AMD
Smarter Choice

Harnessing the Power of Multiple GPUs

Games Developer Conference 2008

Jon Story, jon.story@amd.com
Holger Grün, holger.gruen@amd.com

Agenda

AMD
Smarter Choice

- Why MGPU?
- Driver & Hardware Considerations
- Common Pitfalls & Solutions
- Performance Gains
- Call to Action

AMD
Smarter Choice

Why MGPU?

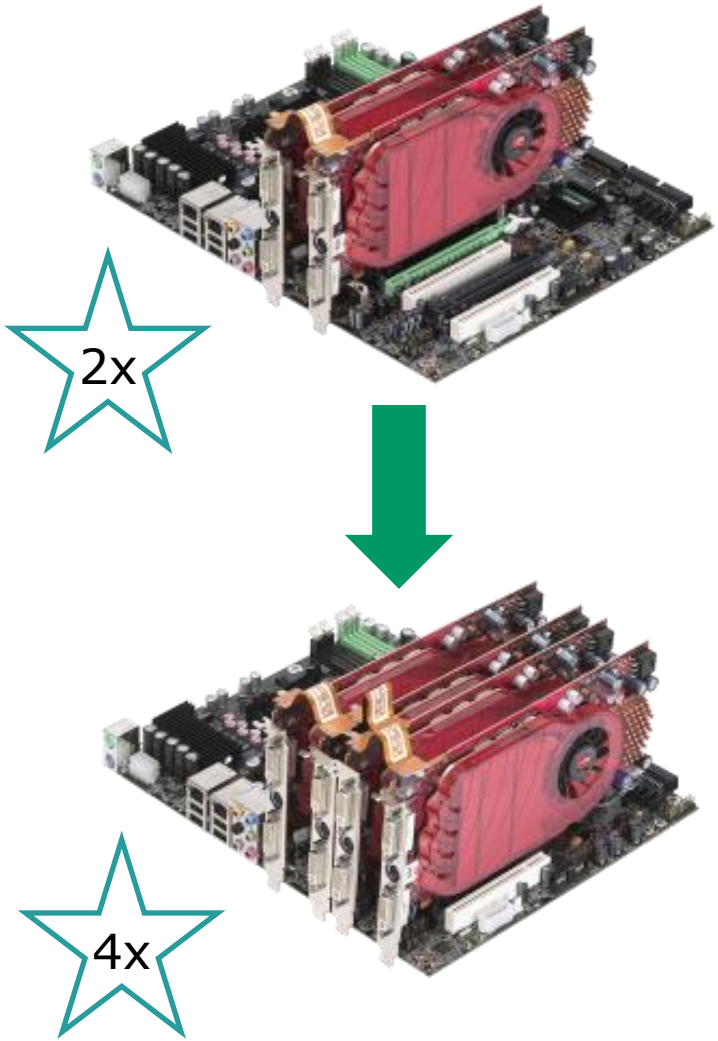
Why MGPU?

AMD
Smarter Choice

- MGPUs can be used to dramatically increase performance and visual quality
  - At higher screen resolutions
  - Especially with increased use of MSAA
- Many applications become GPU limited at higher screen resolutions
  - High resolution monitors => mainstream affordability
- Achieve next generation performance on today‘s HW
  - Prototype your next engine
- Provides an upgrade path for mainstream parts

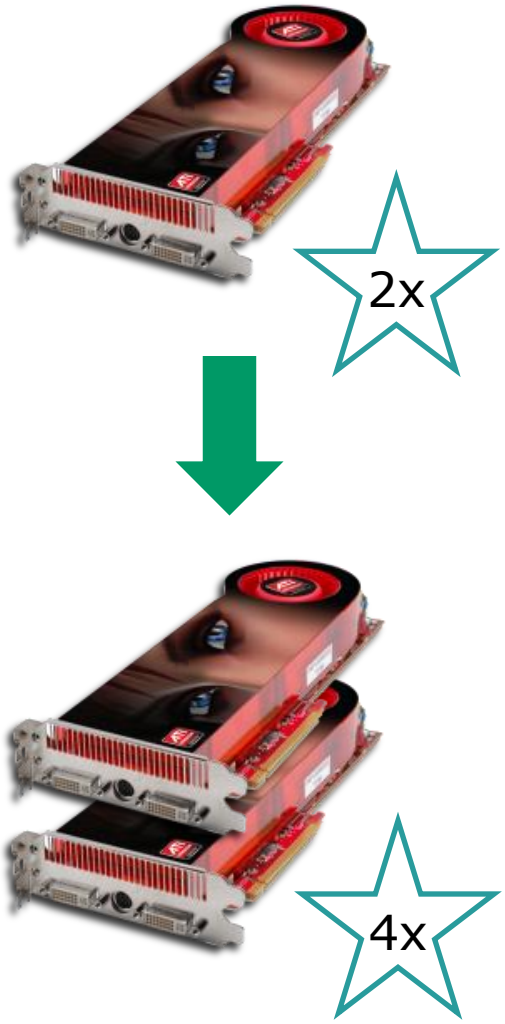
Multiple Boards

- An increasing number of motherboards can accept 2 or more discrete video cards
- Connected by high speed crossover cables
- Now possible to fit 4 Radeon HD3850 boards to a single motherboard
- CrossFireX technology allows you to harness that performance



Multiple GPUs per Board

- The Radeon HD3870 X2 is a single-board multi-GPU architecture
  - AFR is on by default
- Heavy peer to peer communication
  - Bi-directional 16x lane pipe connecting the 2 GPUs
- CrossFireX supports 2 HD3870 X2 boards for Quad GPU performance



Hybrid Crossfire

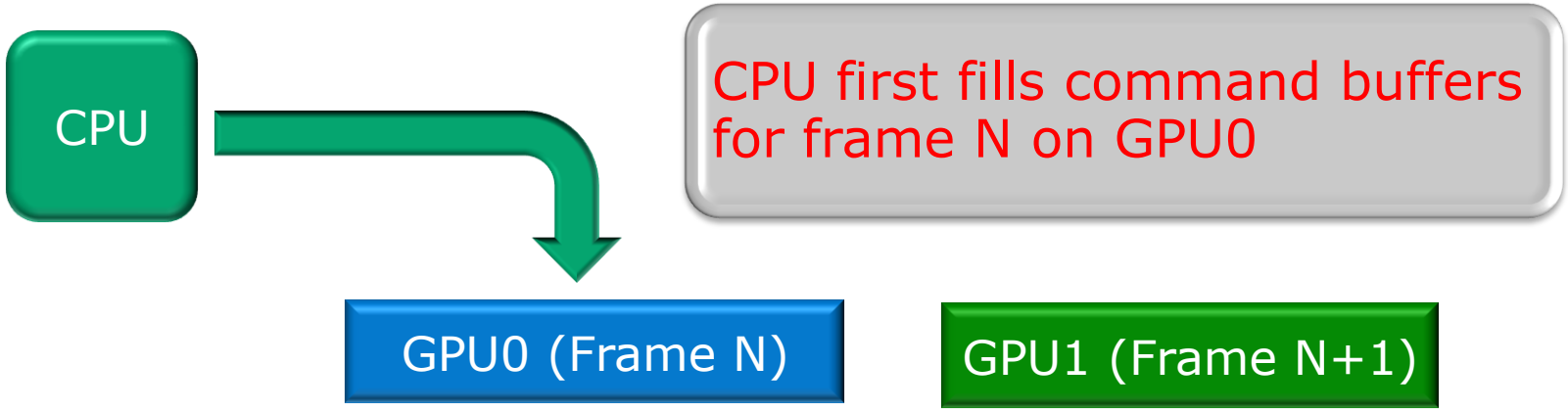
- Combination of integrated and discrete graphics
- 3D graphics performance boost
  - Laptops
  - Mainstream desktop PCs
- Use less power during non-taxing graphical tasks

CrossFire Rendering Modes

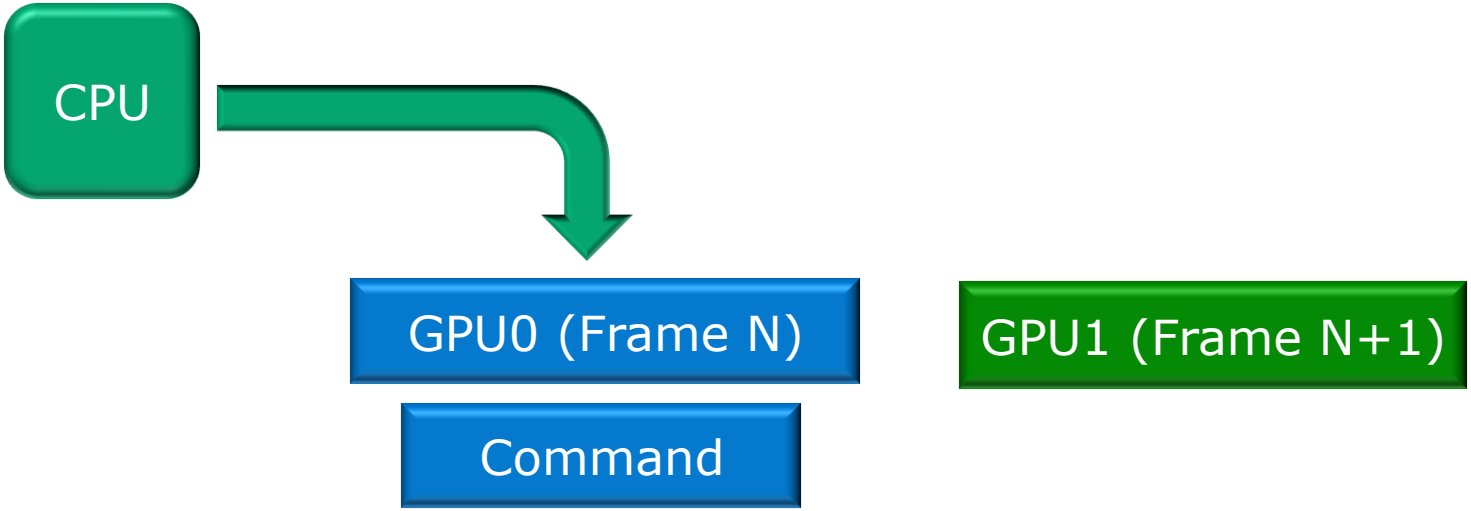
- Split Frame Rendering / Scissor
  - Screen is divided into number of GPUs
  - Dynamic load balancing

- Alternate Frame Rendering
  - GPUs take alternate frames
  - Vertex processing not duplicated
  - Highest performing mode

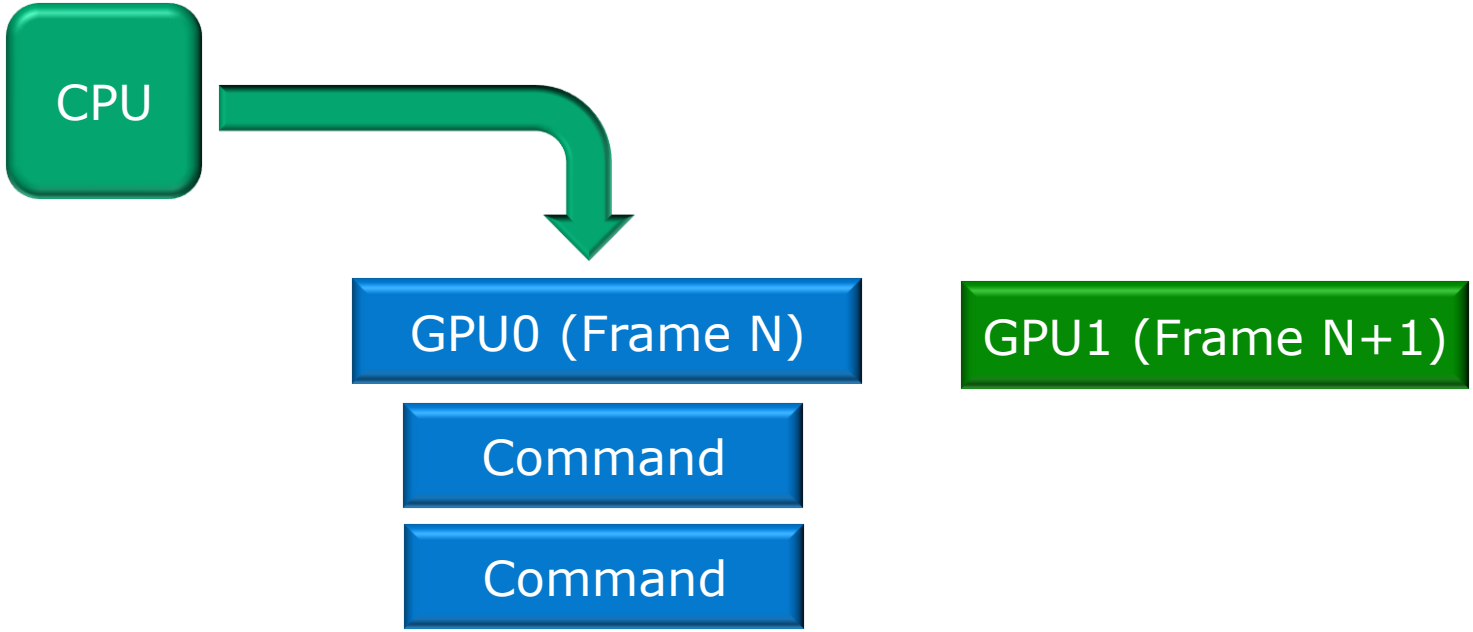
How does AFR Work?



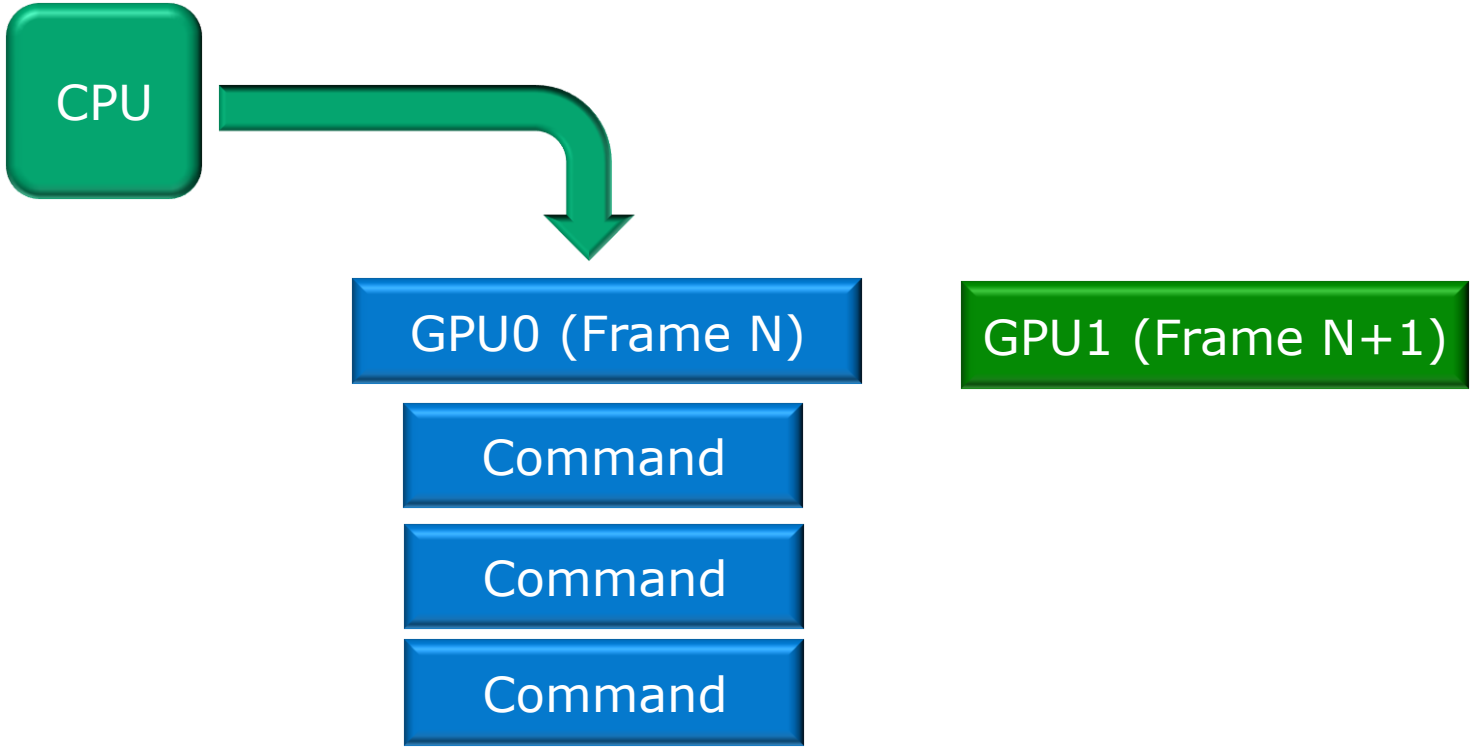
How does AFR Work?



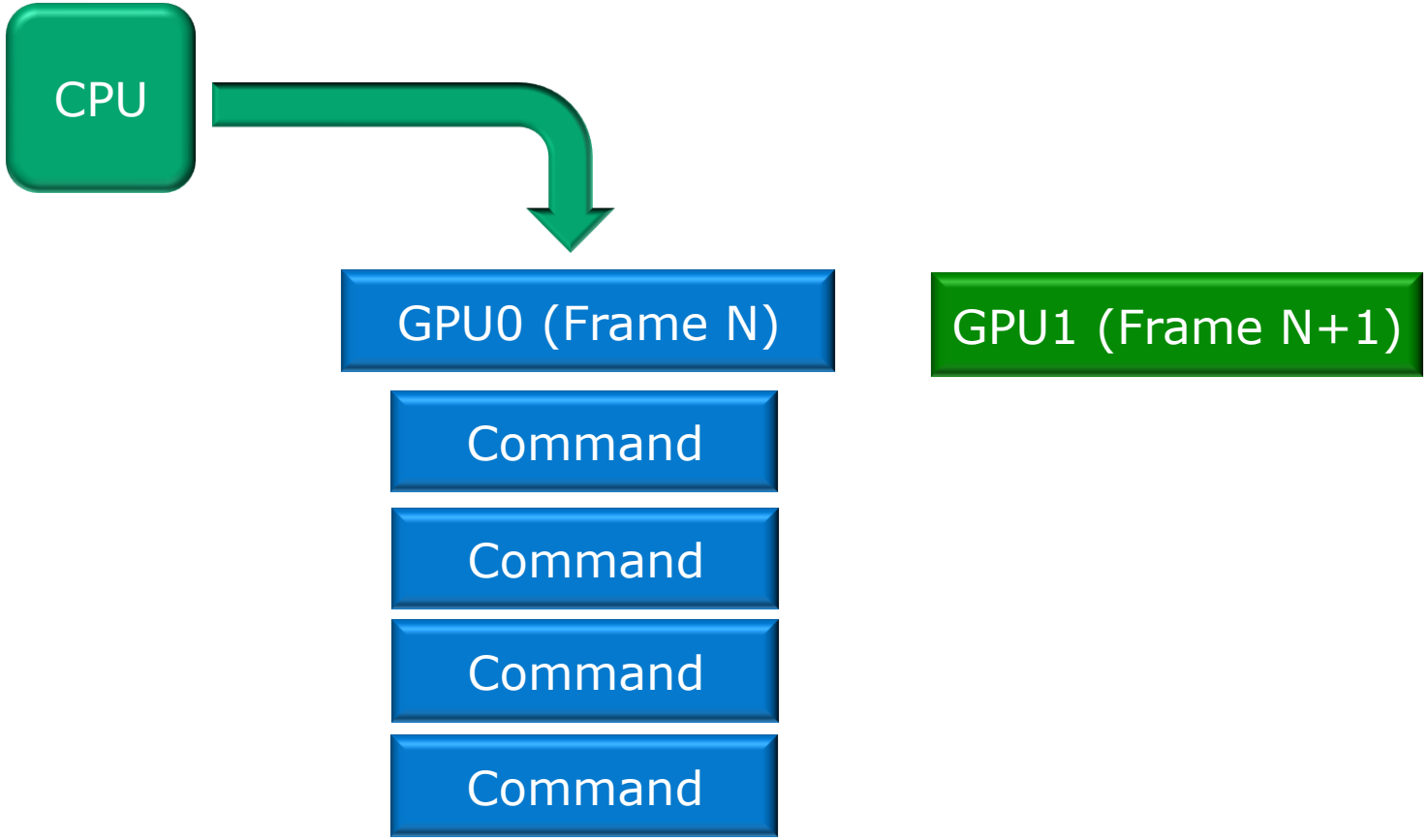
How does AFR Work?



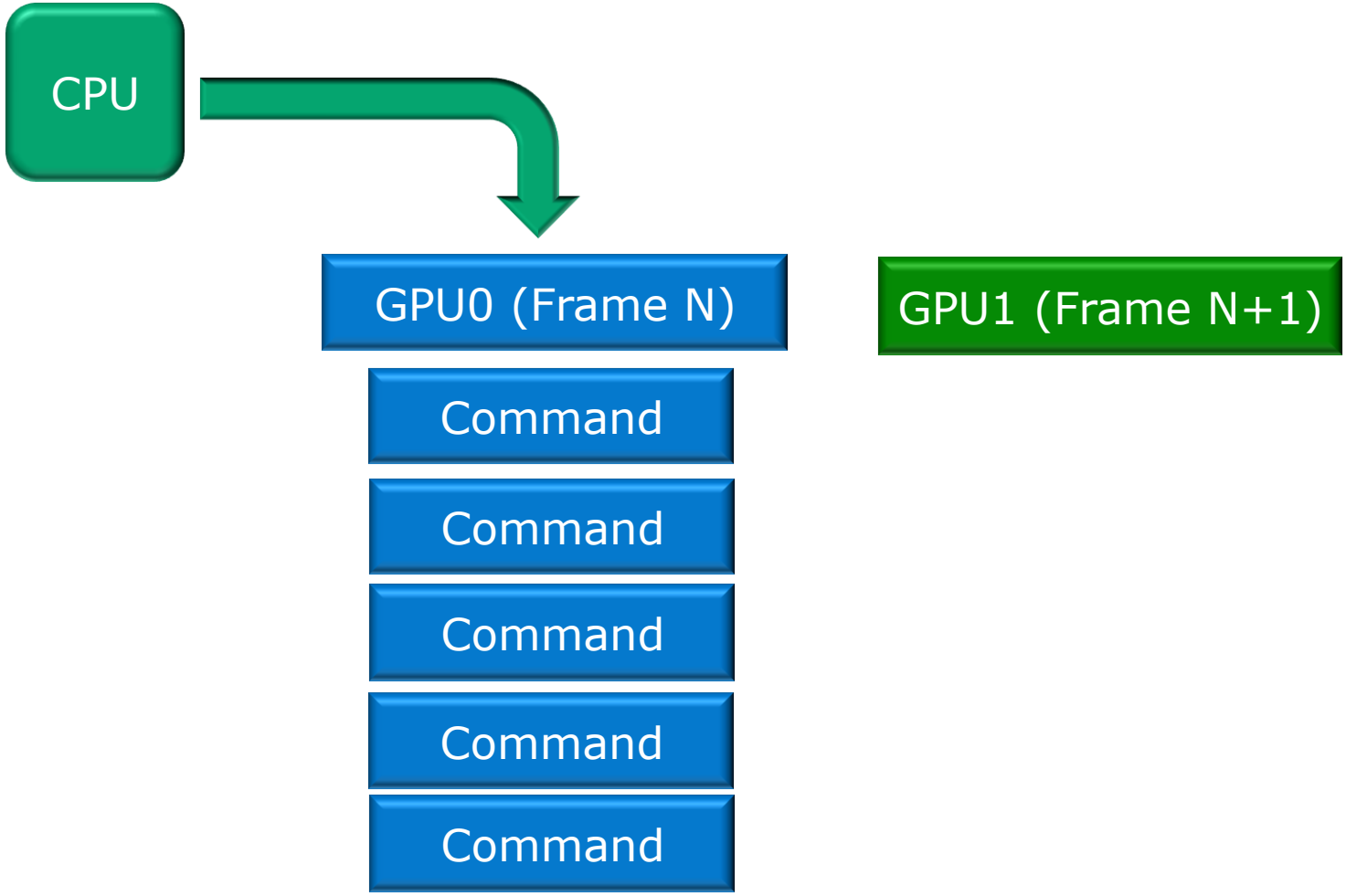
How does AFR Work?



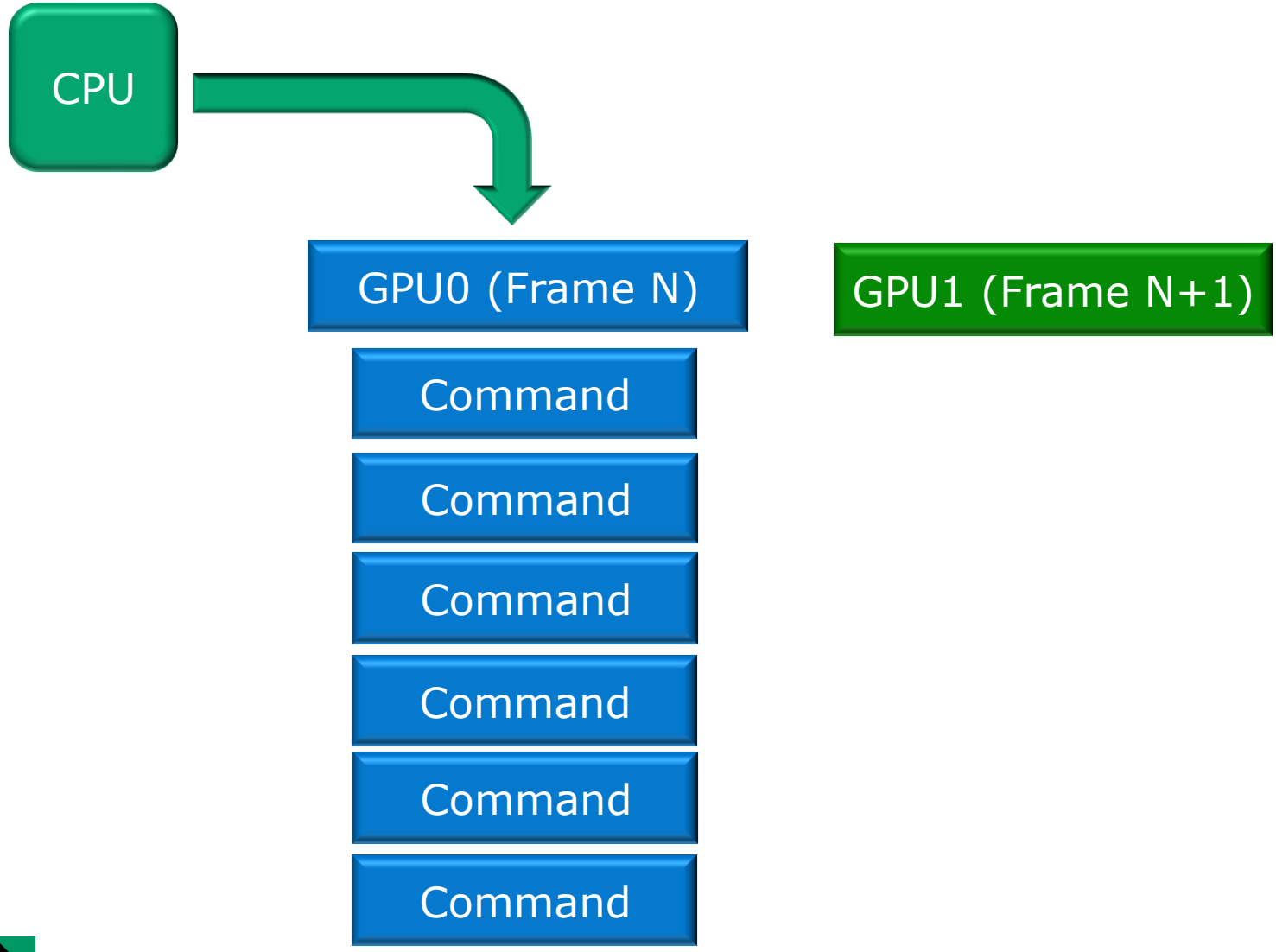
How does AFR Work?



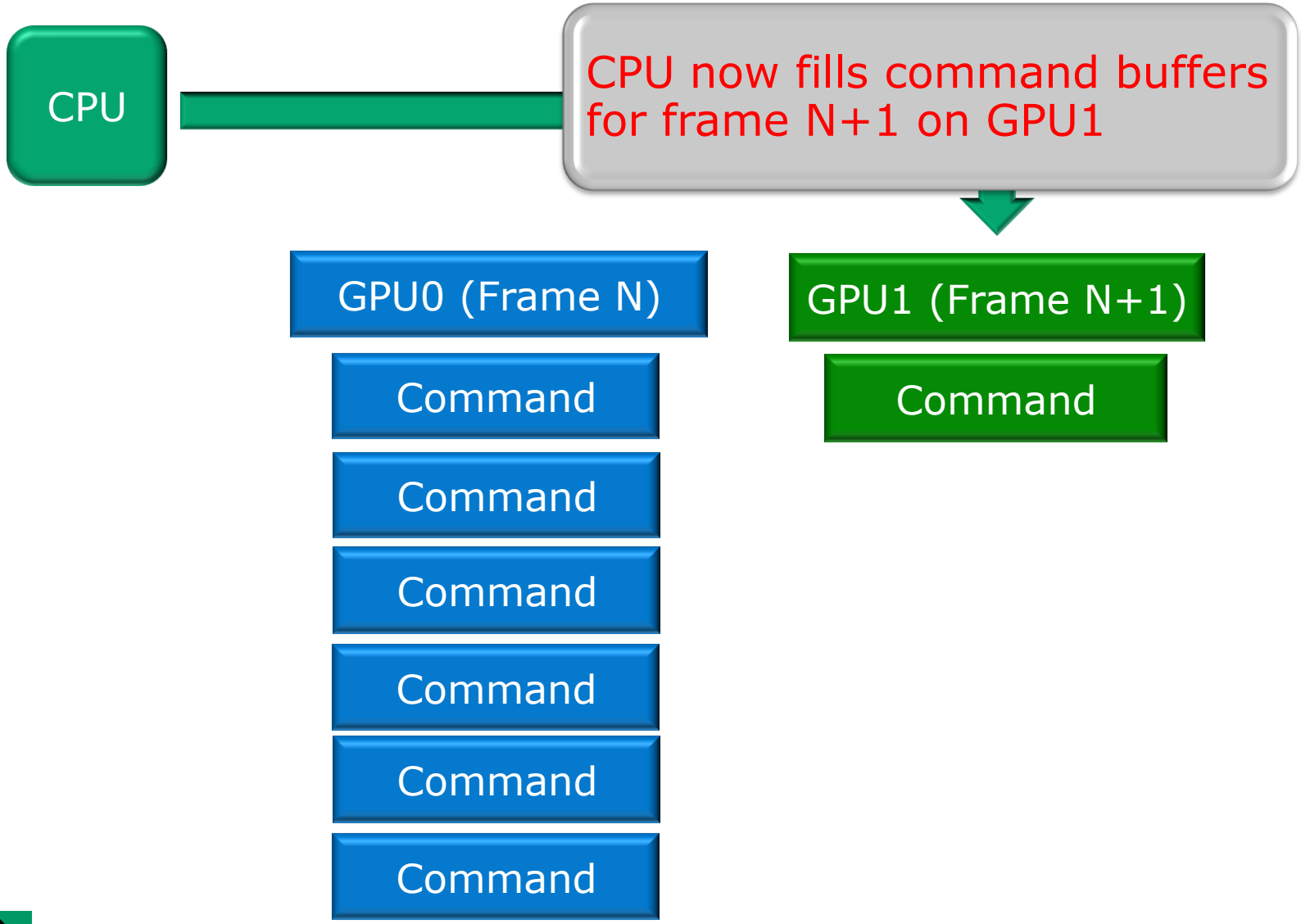
How does AFR Work?



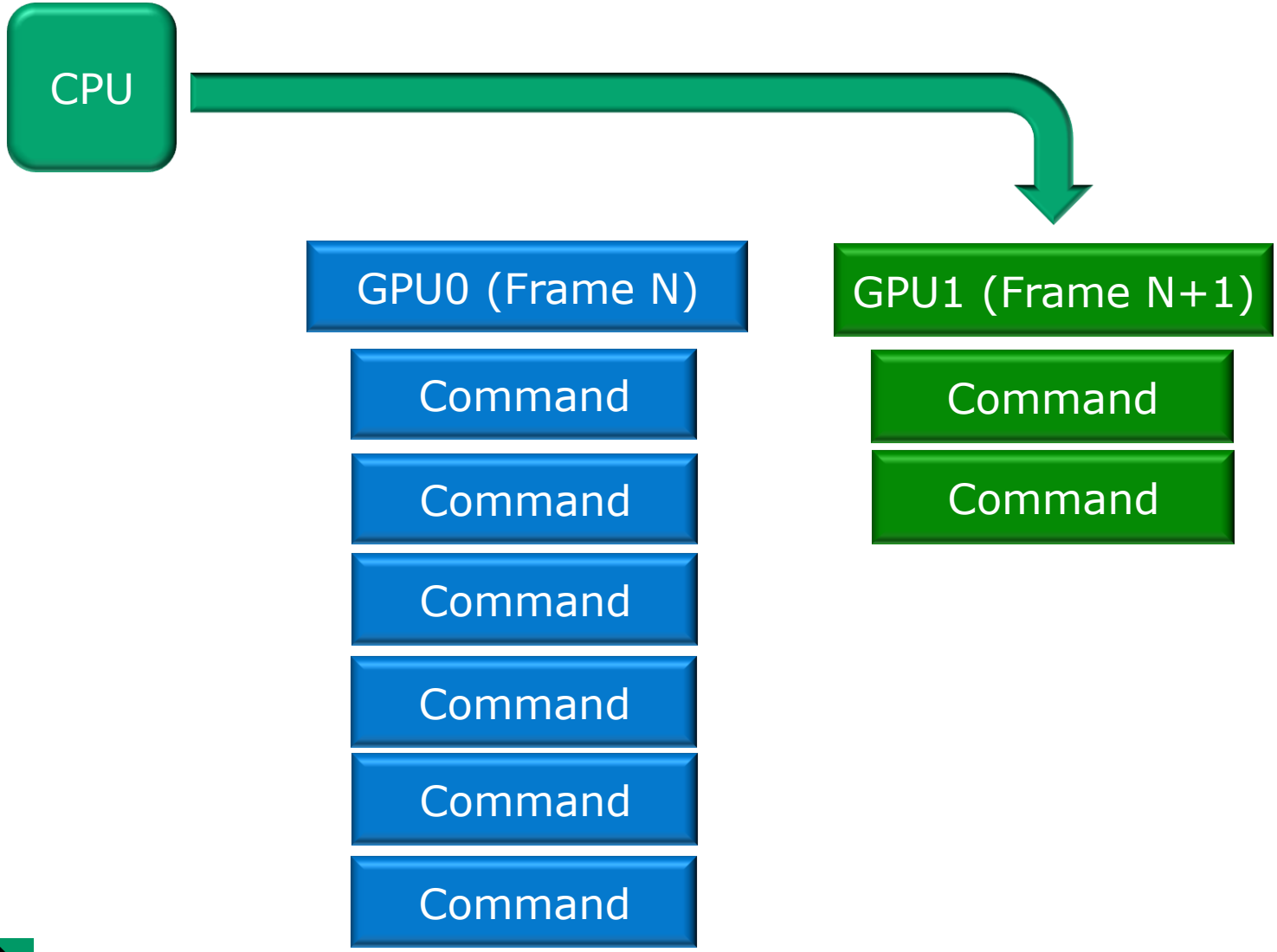
How does AFR Work?



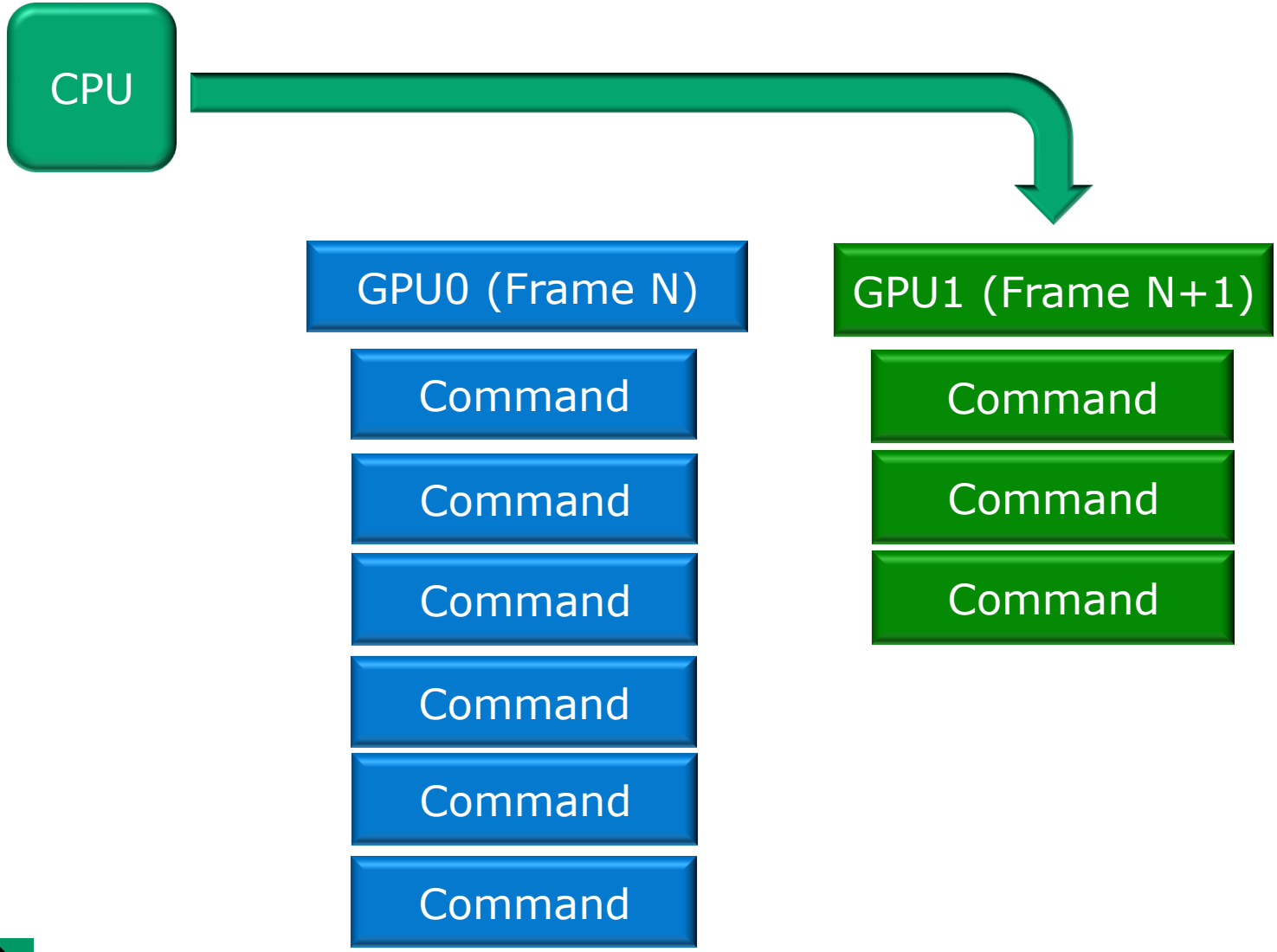
How does AFR Work?



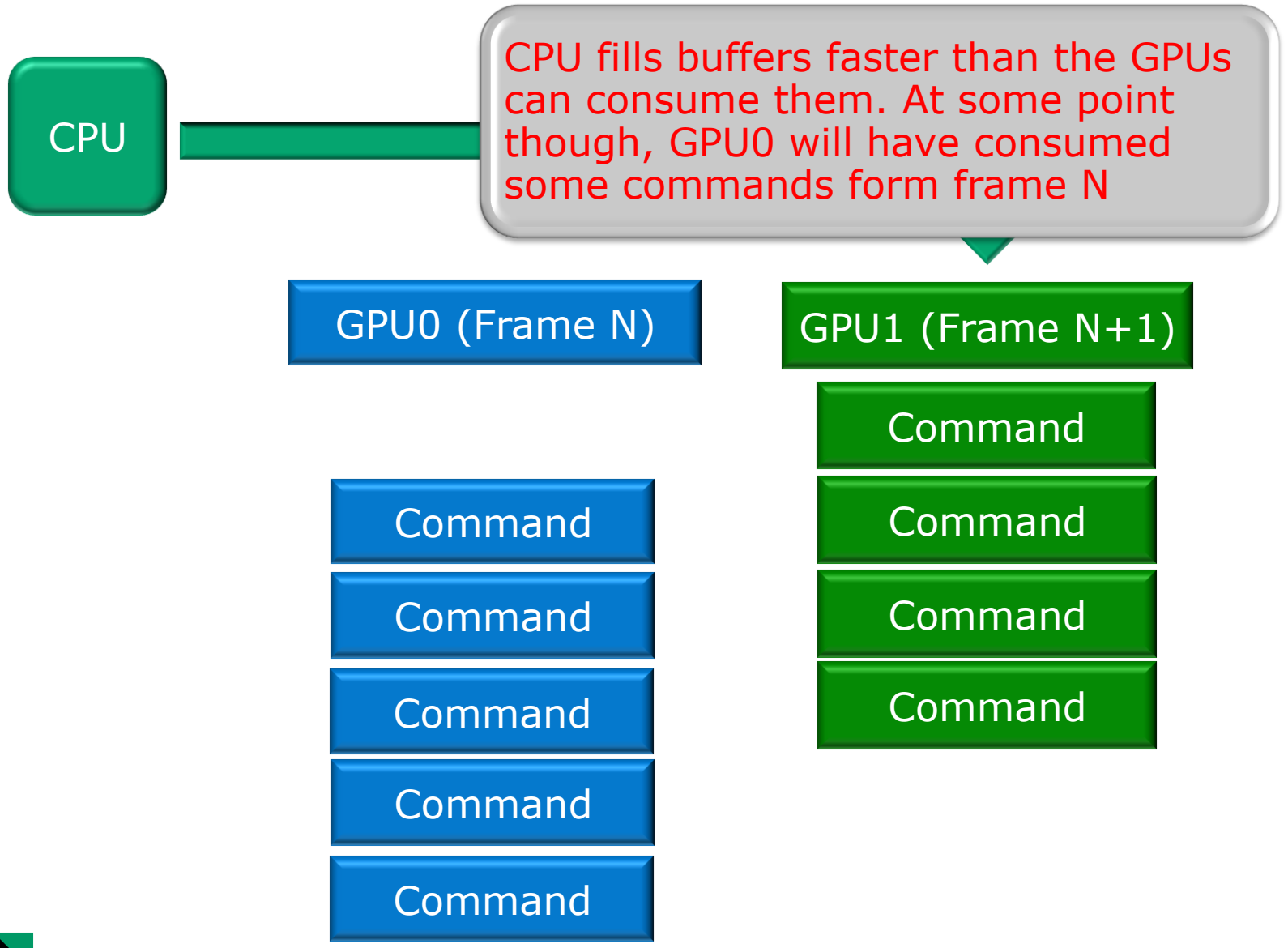
How does AFR Work?



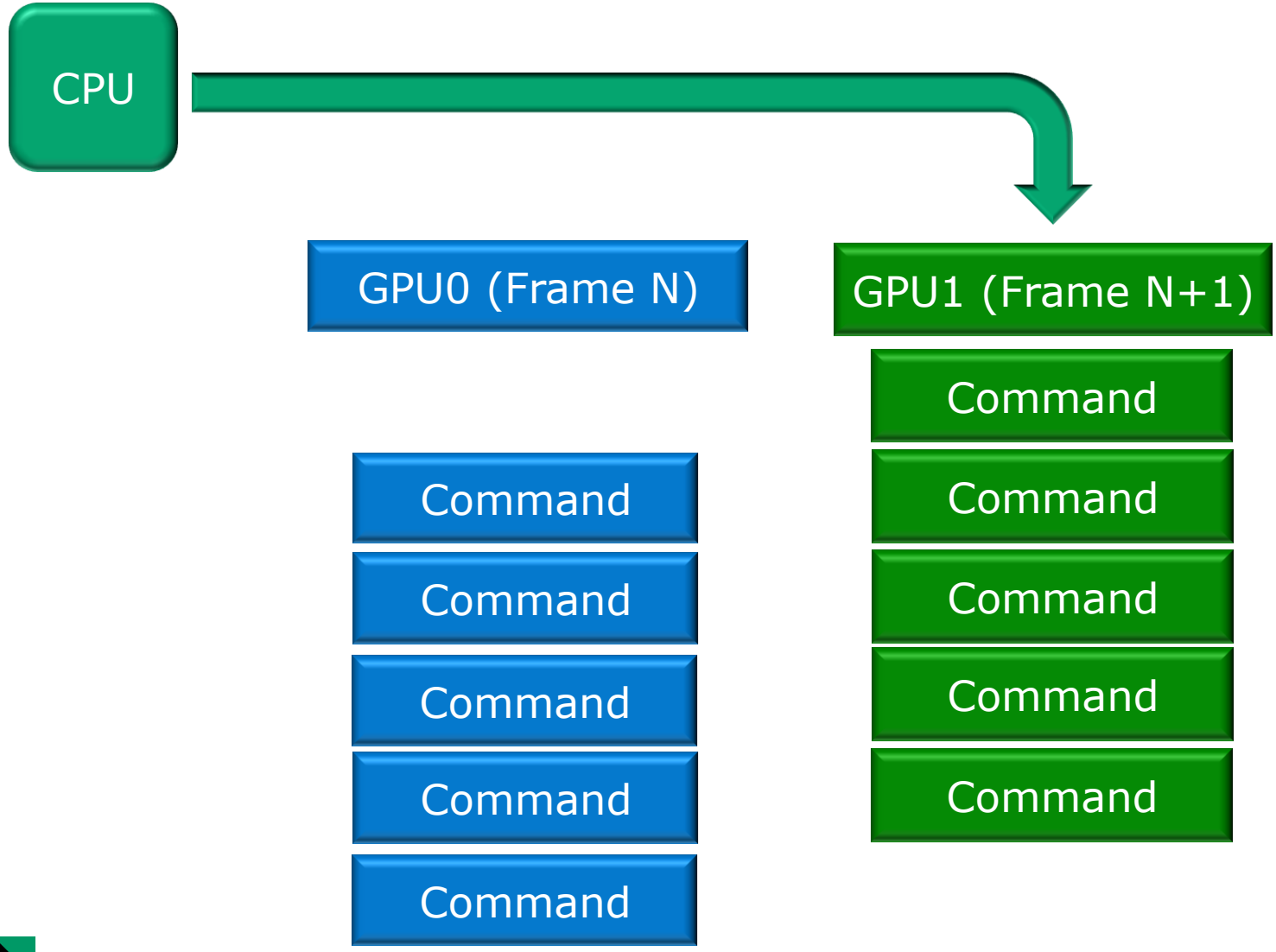
How does AFR Work?



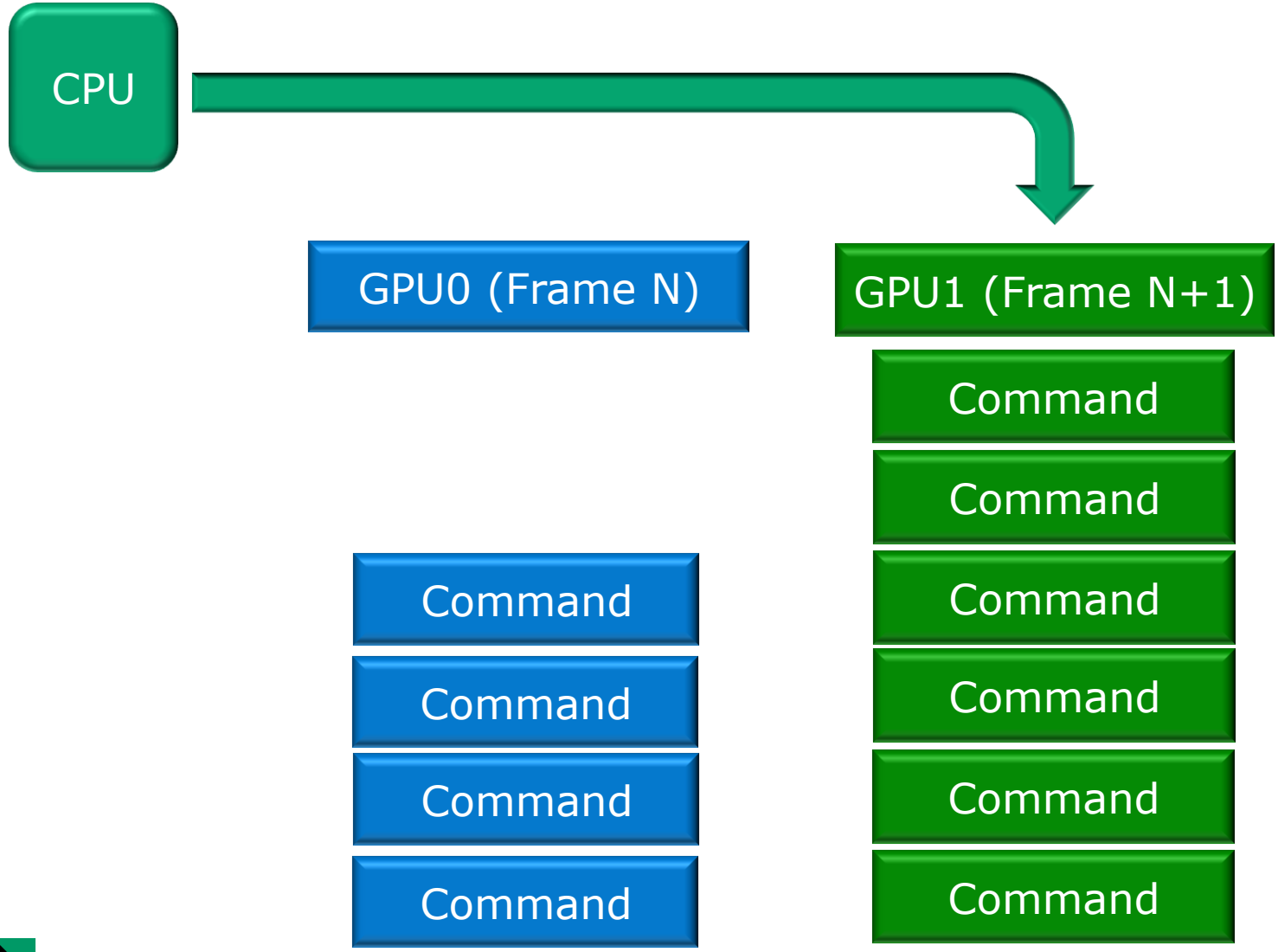
How does AFR Work?



How does AFR Work?



How does AFR Work?



Driver & Hardware Considerations

Hardware Considerations

AMD
Smarter Choice

- Current MGPU setups are not shared memory architectures
  - Resources placed in local video memory are duplicated for each GPU
- Driver initiates peer to peer (P2P) copies to keep resources in sync
  - On some chipsets this may involve the CPU
  - Synchronizes all GPUs
  - Very heavy impact on performance that can even result in negative scaling

Driver Modes

- Compatible AFR Mode
  - Default mode
  - Driver checks for AFR unfriendly behaviour
  - Will peer to peer (P2P) copy stale resources
- Full AFR Mode (Application Profile)
  - Driver recognises EXE name
    - Use a unique name and don‘t change it
  - Behaviour fully guided by profile
  - Best performance – no checking
  - Rename EXE to “AFR-FriendlyD3D.exe“
    - Use “AFR-FriendlyOGL.exe“ for OpenGL
    - No checking : Speed & compatibility test

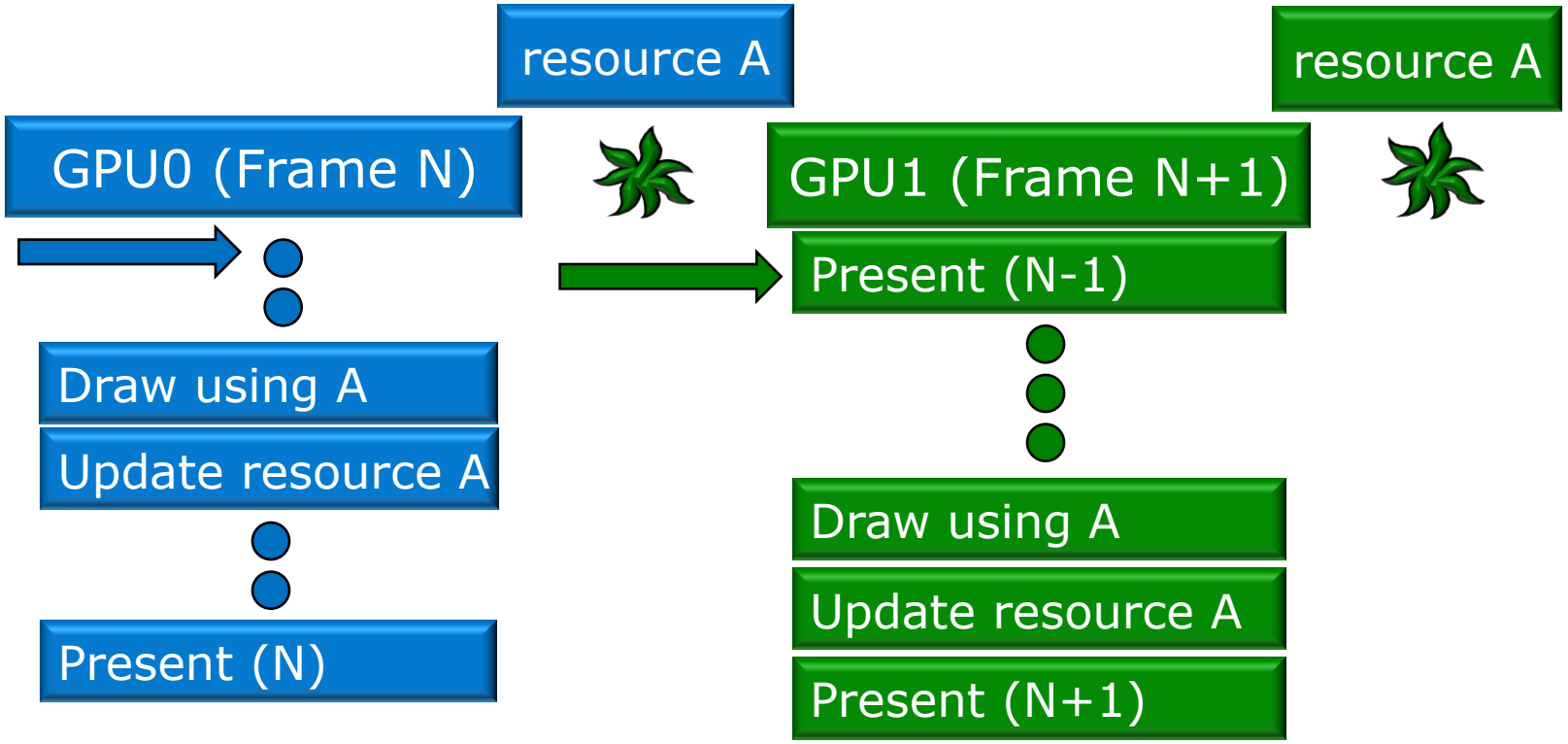
Detecting the Number of GPUs

AMD Smarter Choice

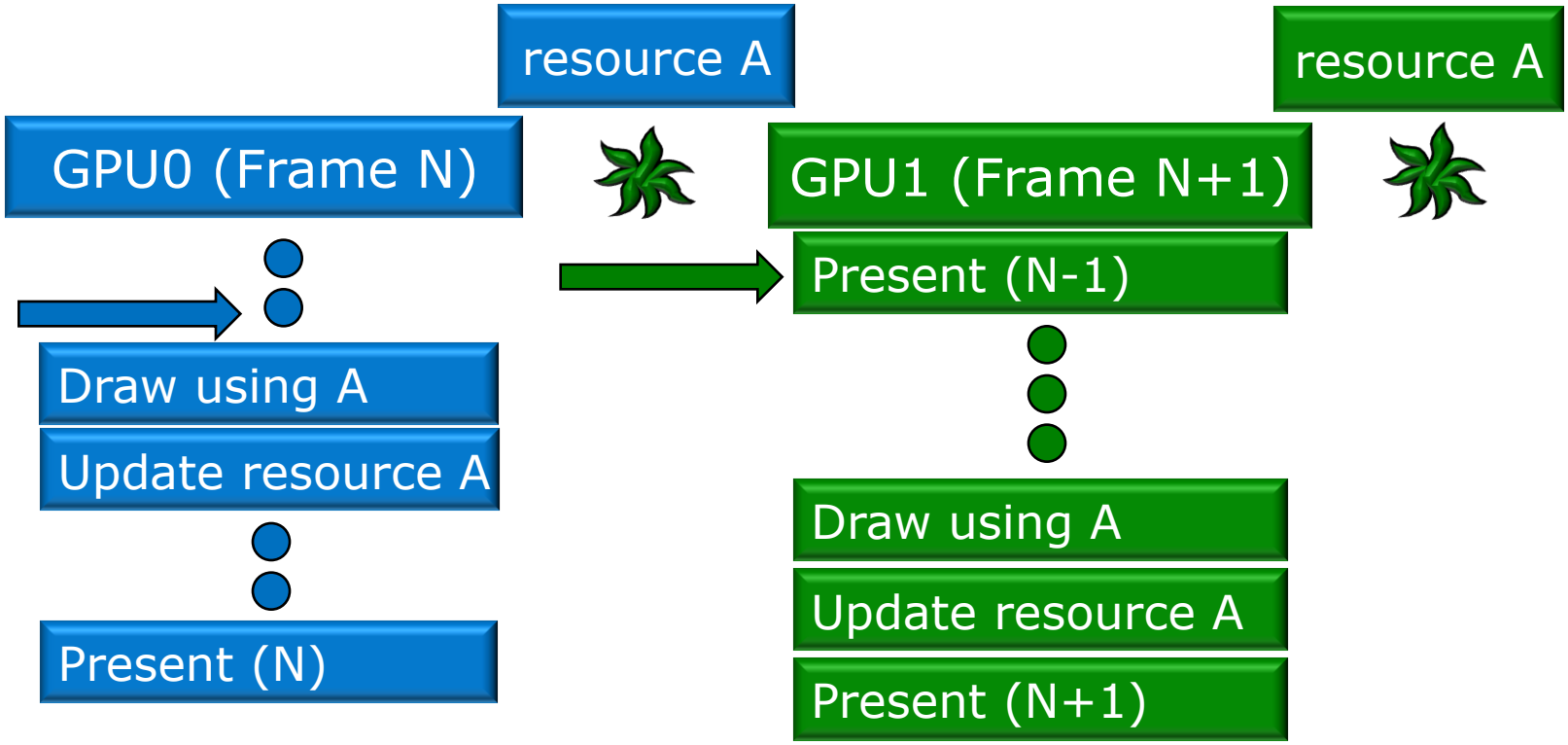
- Visit http://ati.amd.com/developer
  - Download project called "CrossFire Detect"
- Statically link to:
  - "atimgpud_s_x86.lib" 32 bit version
  - "atimgpud_s_x64.lib" 64 bit version
- Include header file:
  - "atimgpud.h"
- Call this function:
  - INT count = AtiMultiGPUAdapters();

Common Pitfalls & Solutions

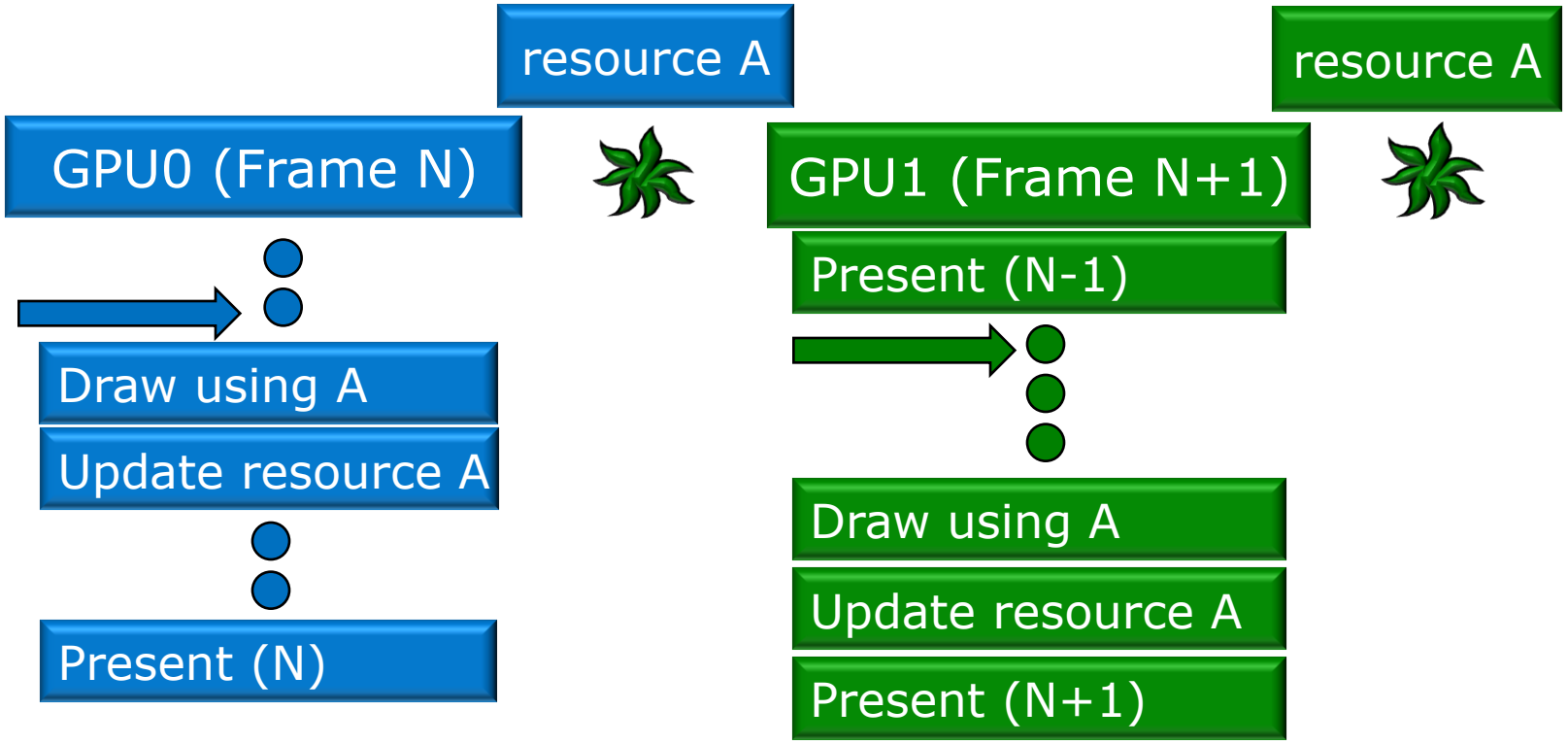
Pitfall: Dependencies Between Frames



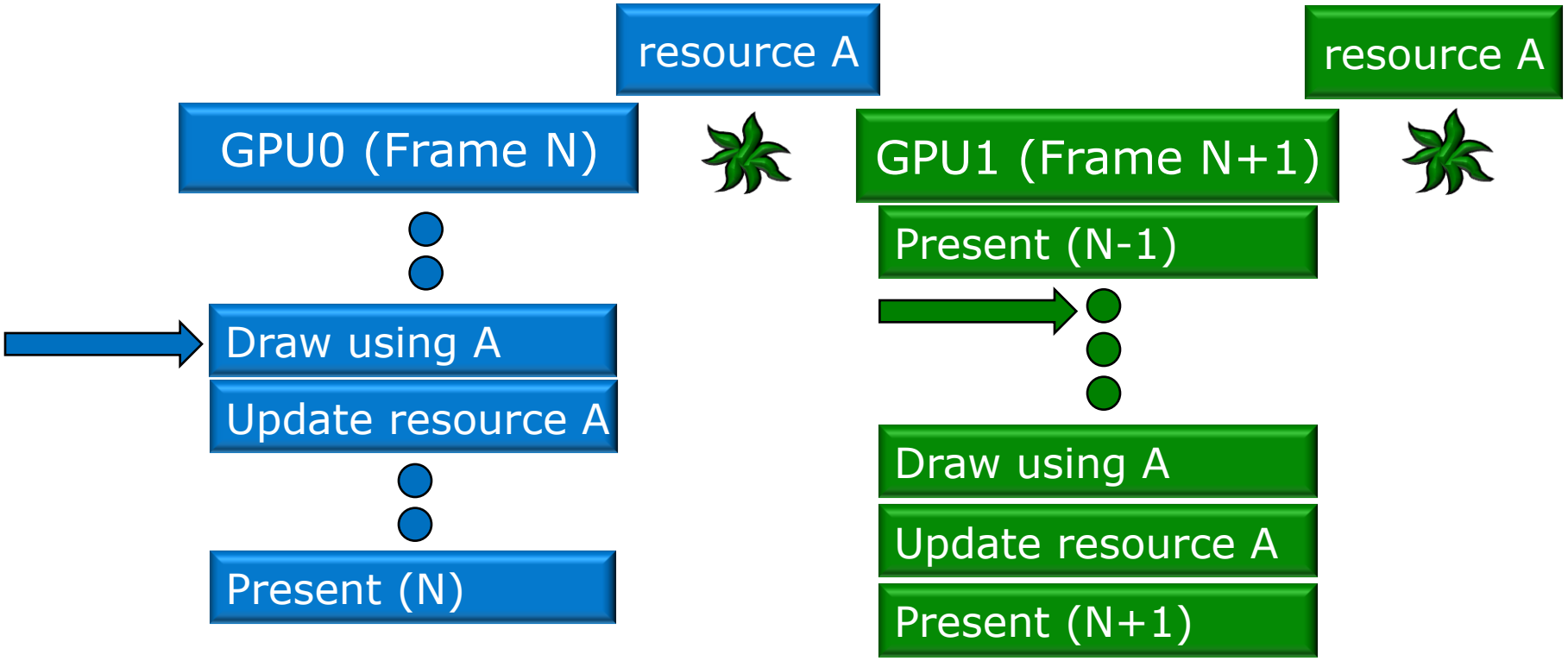
Pitfall: Dependencies Between Frames



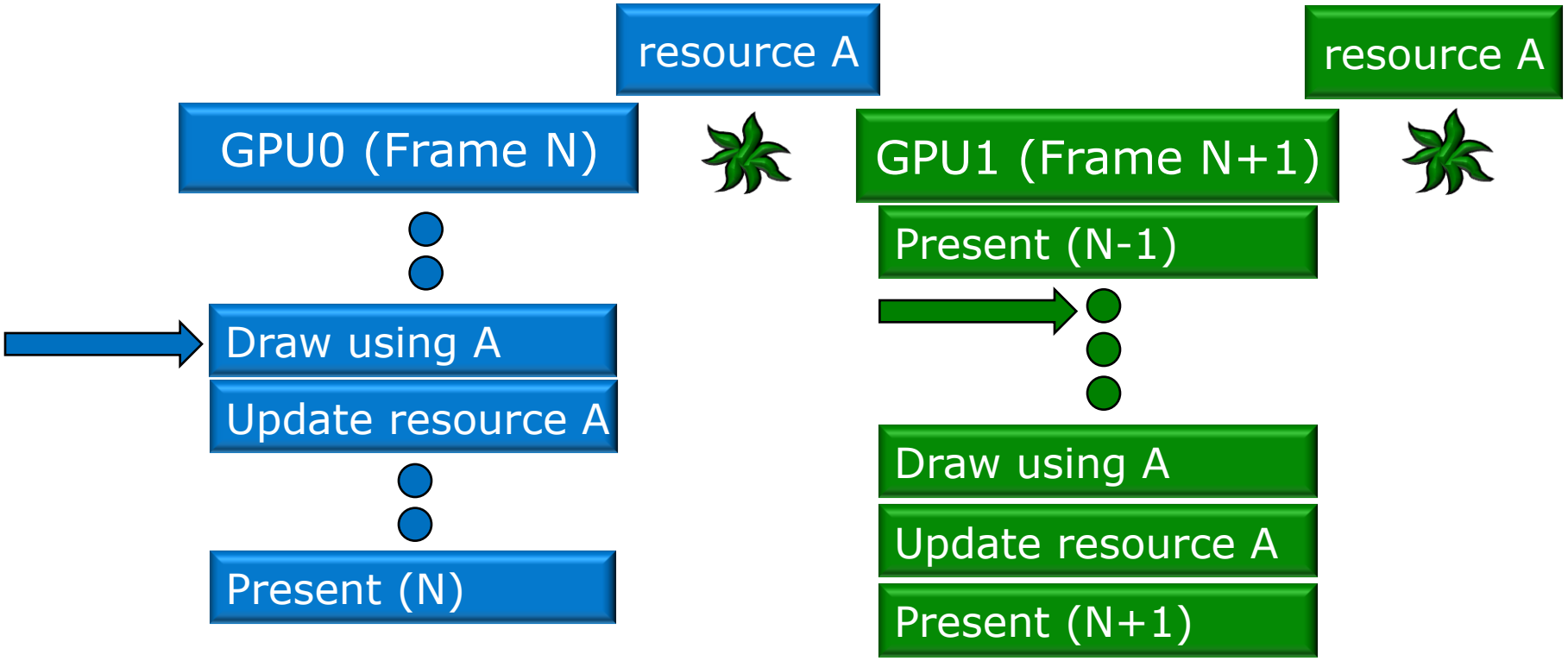
Pitfall: Dependencies Between Frames



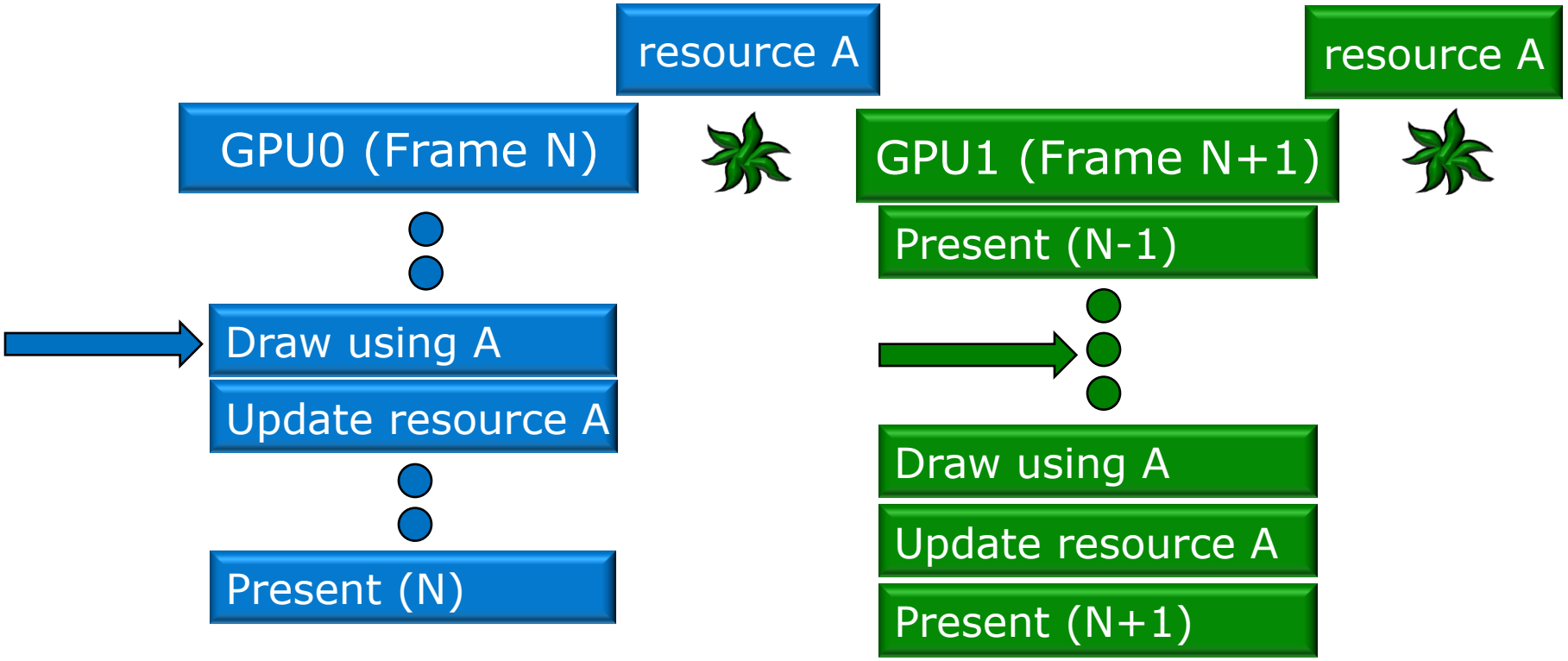
Pitfall: Dependencies Between Frames



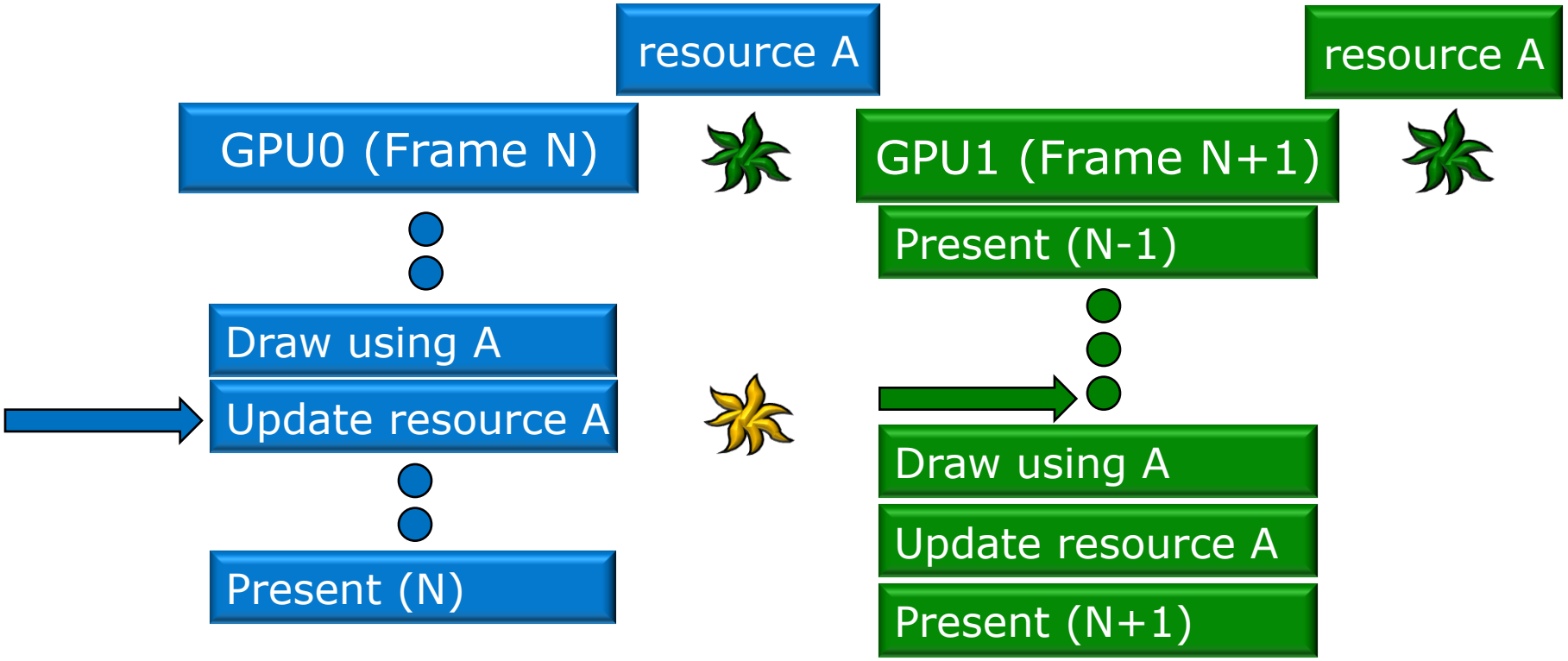
Pitfall: Dependencies Between Frames



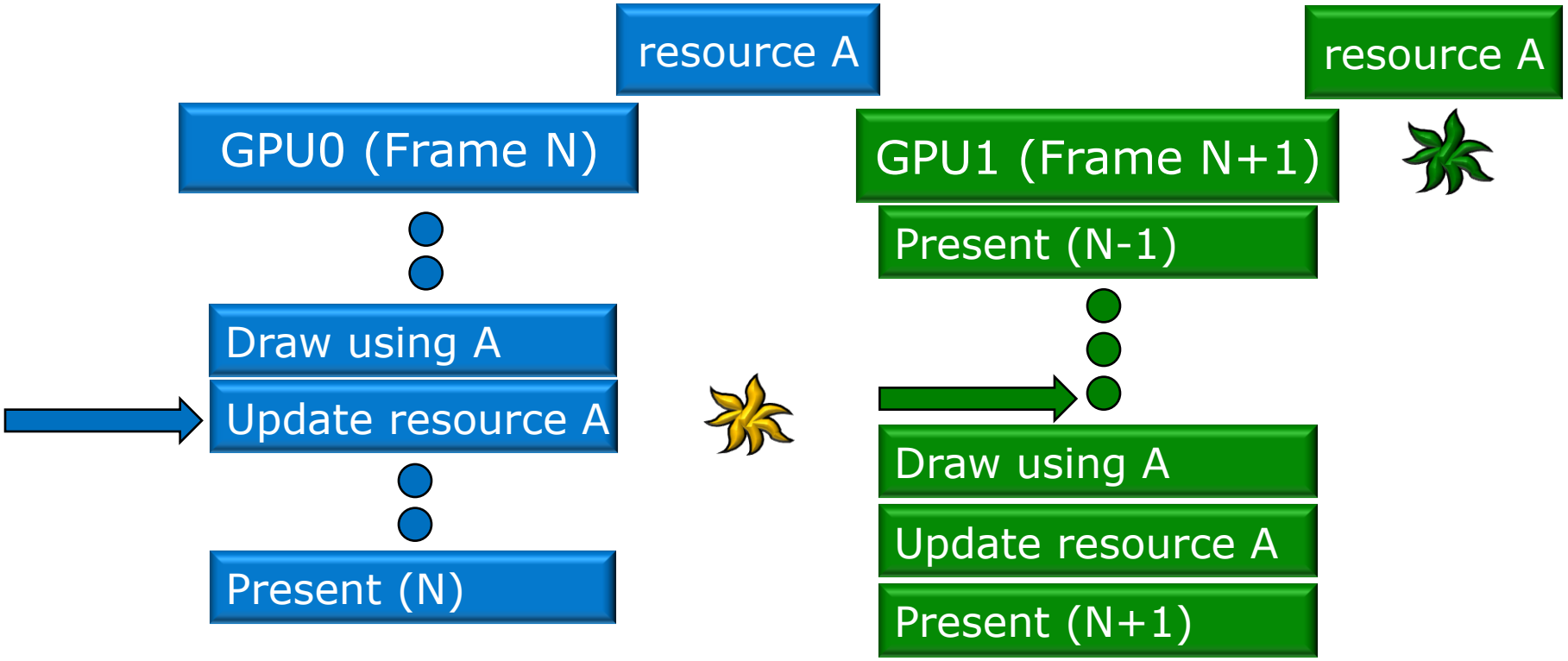
Pitfall: Dependencies Between Frames



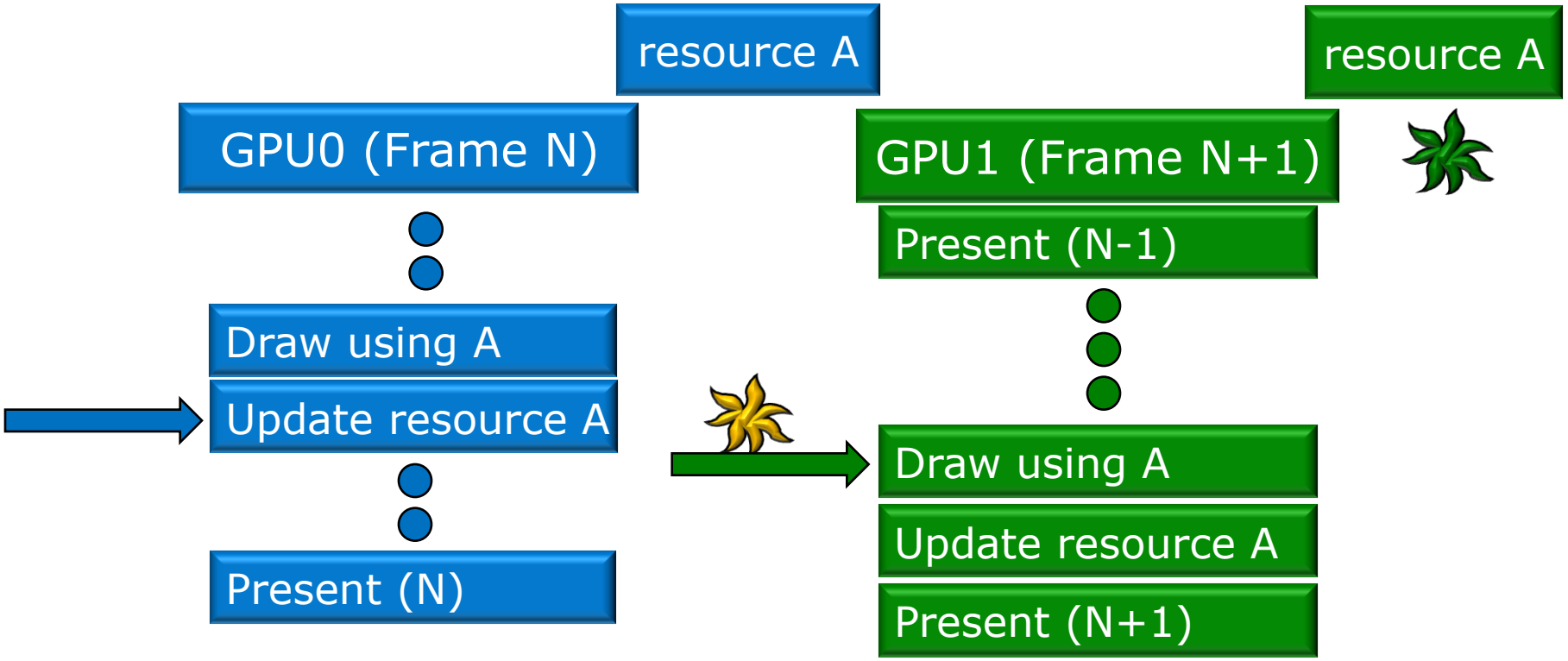
Pitfall: Dependencies Between Frames



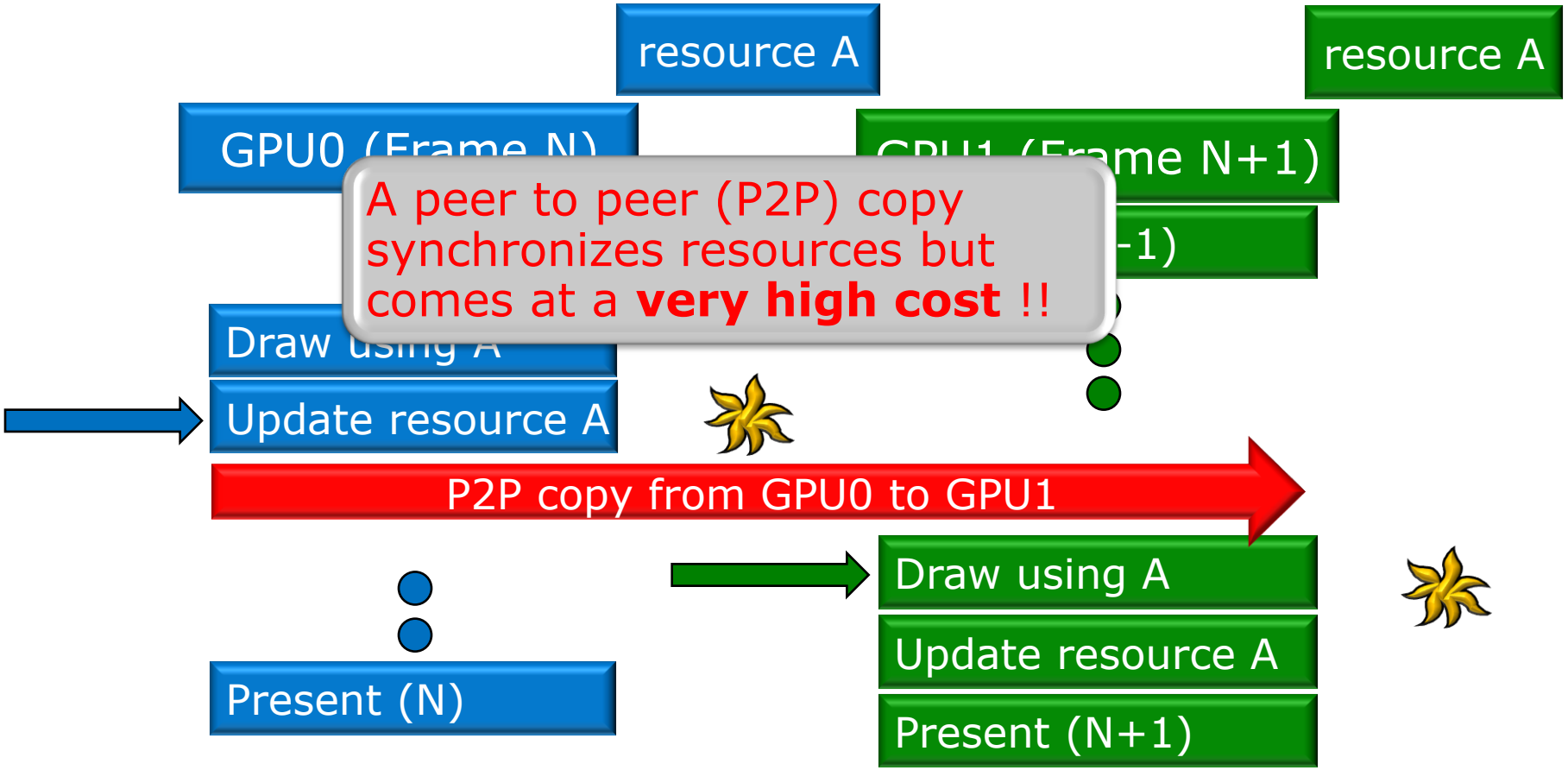
Pitfall: Dependencies Between Frames



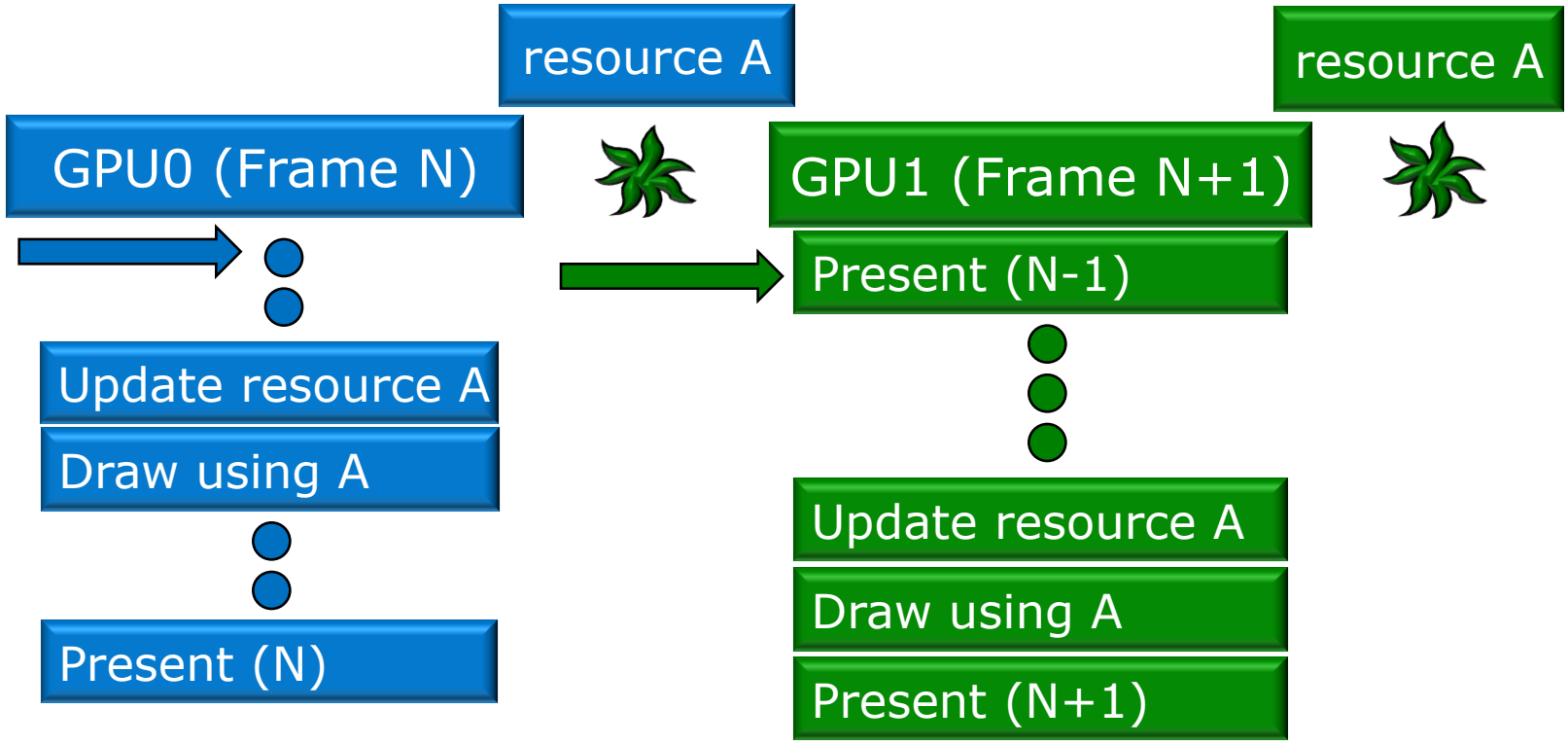
Pitfall: Dependencies Between Frames



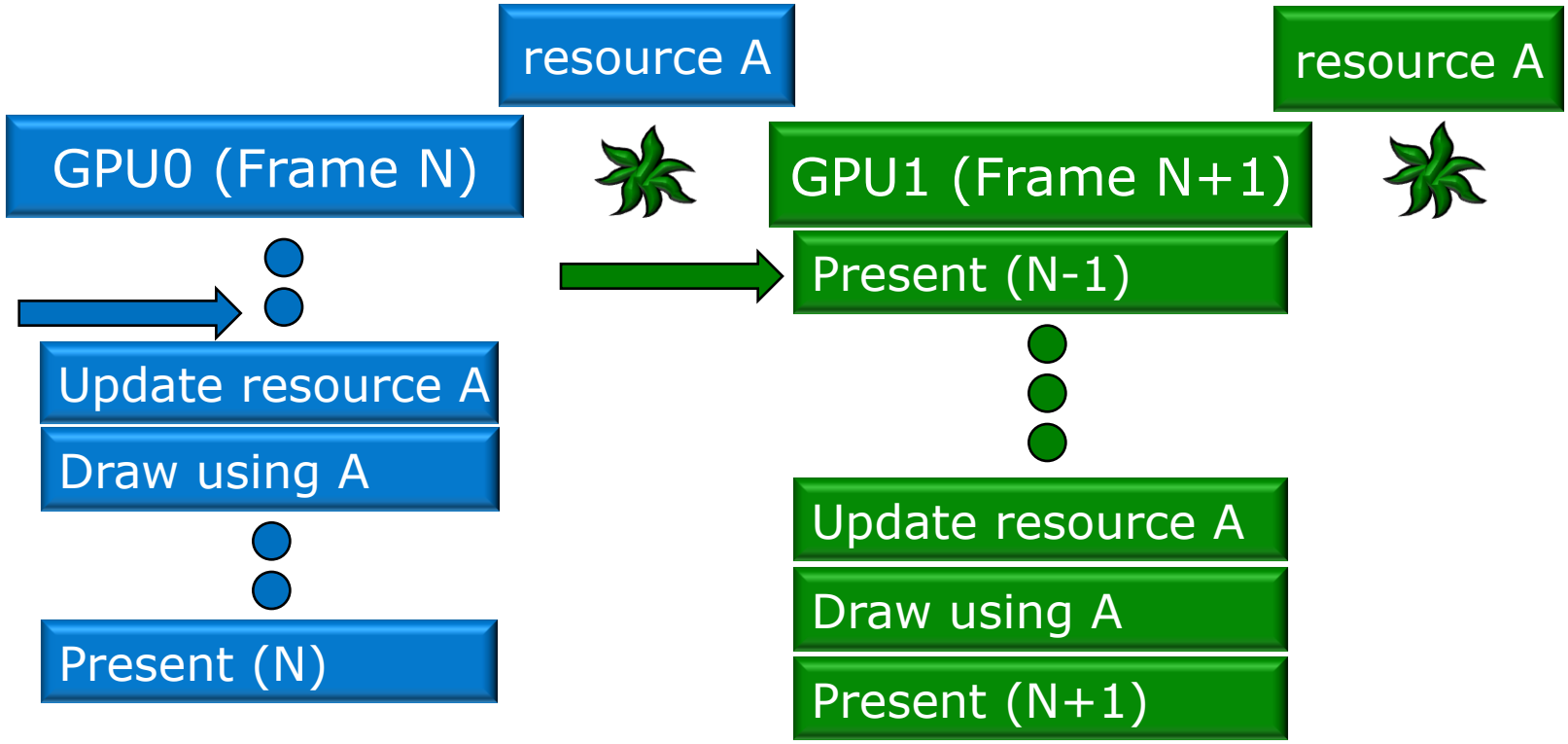
Pitfall: Dependencies Between Frames



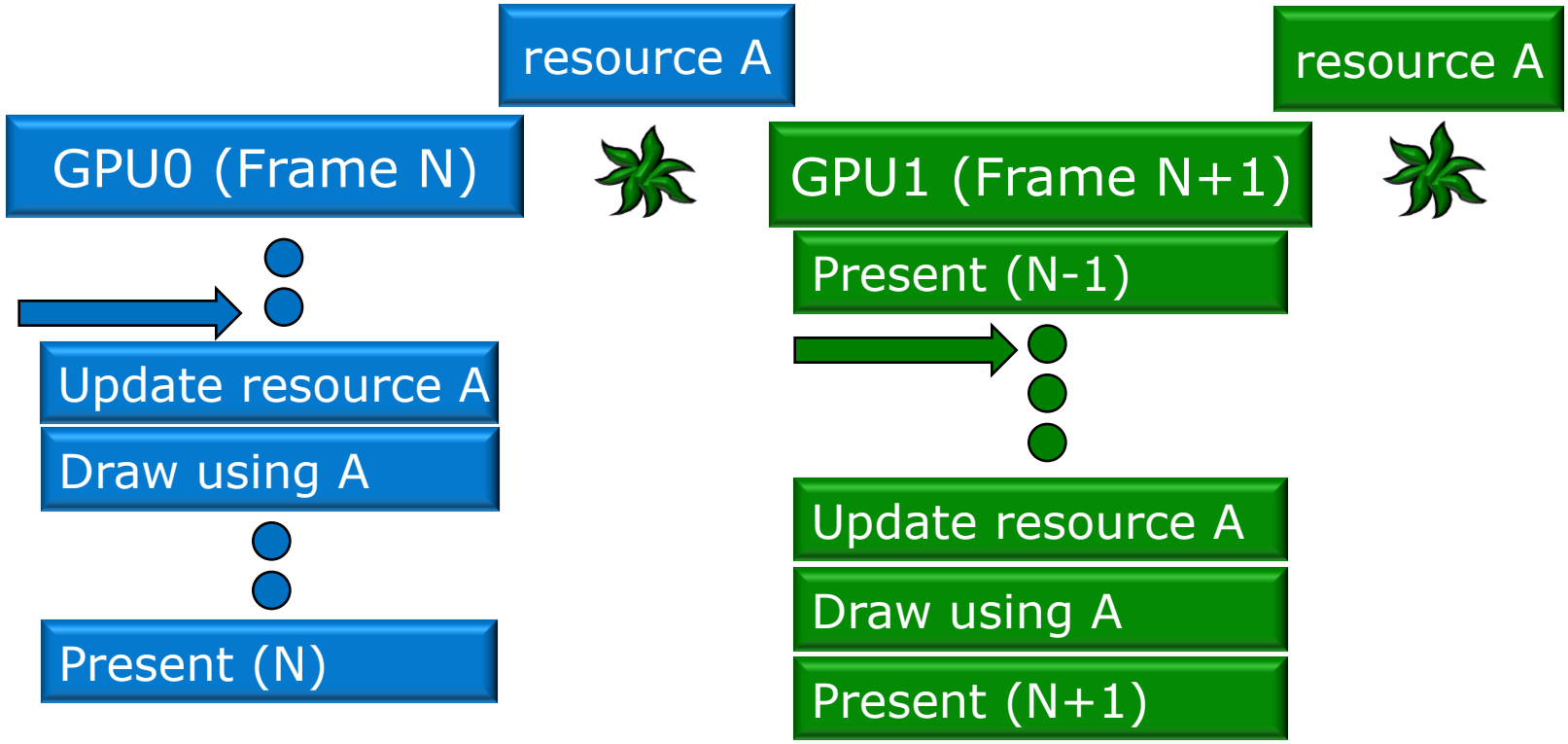
Solution: Resources that Change Every Frame



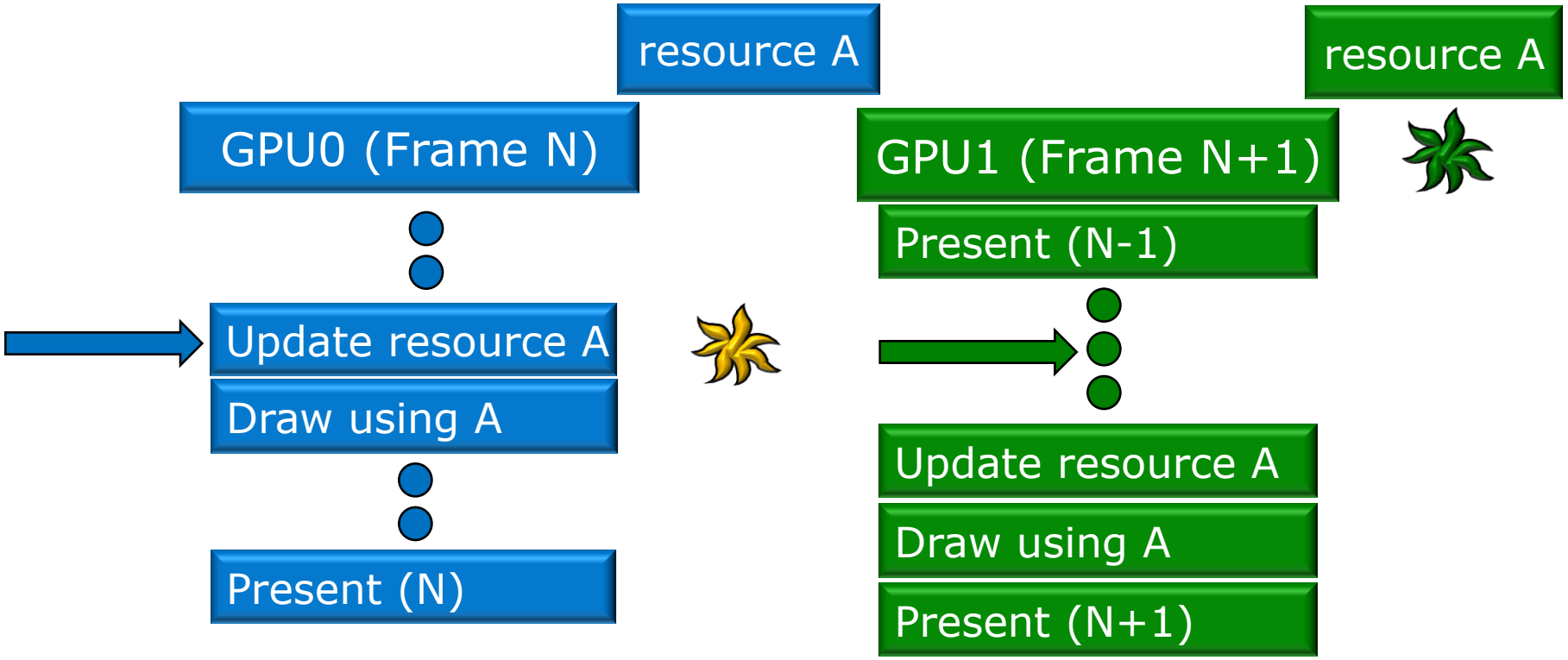
Solution: Resources that Change Every Frame



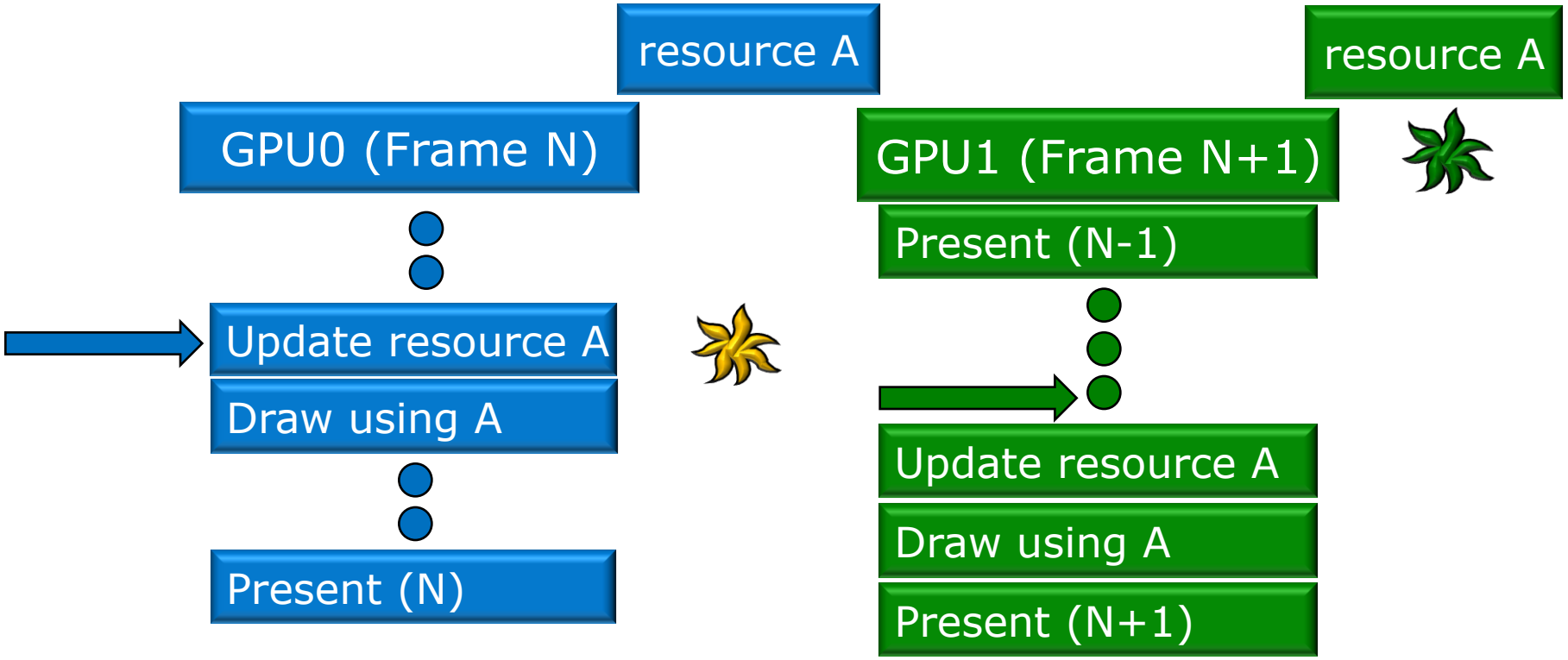
Solution: Resources that Change Every Frame



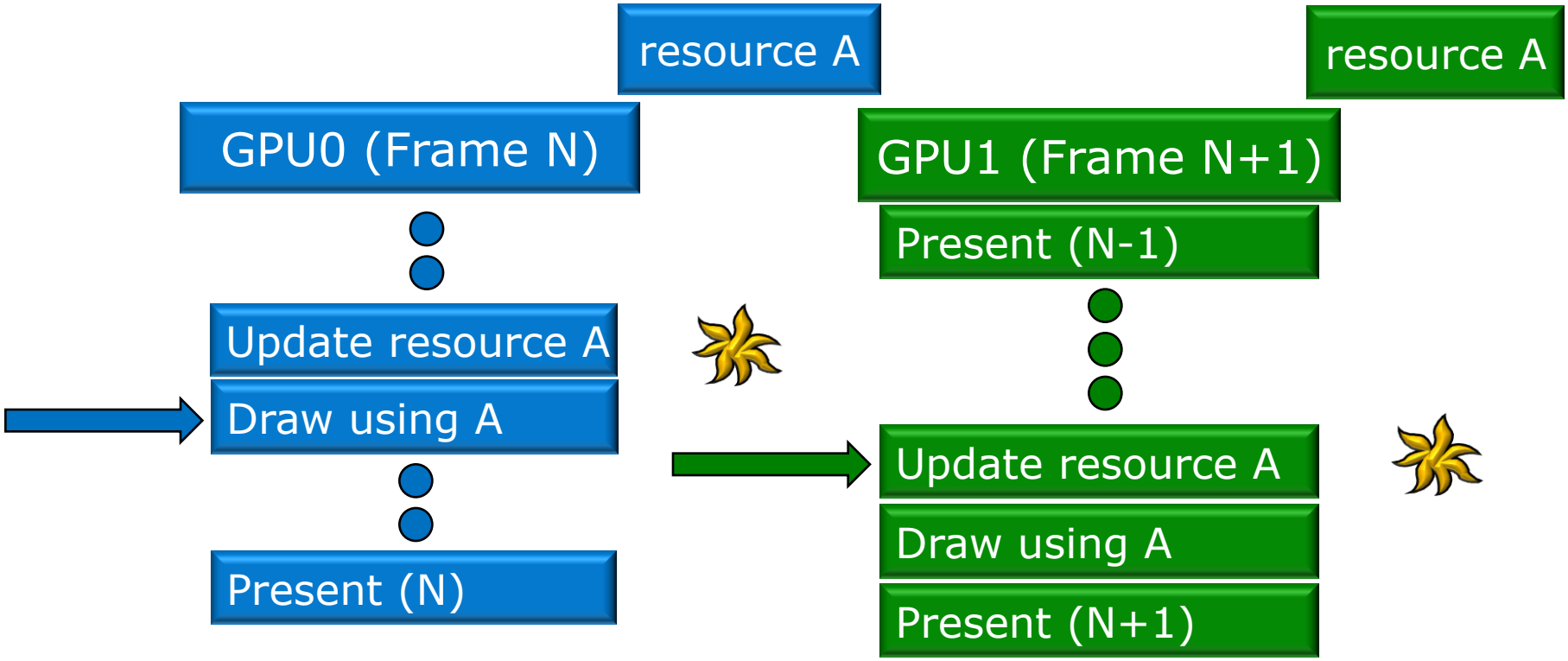
Solution: Resources that Change Every Frame



Solution: Resources that Change Every Frame

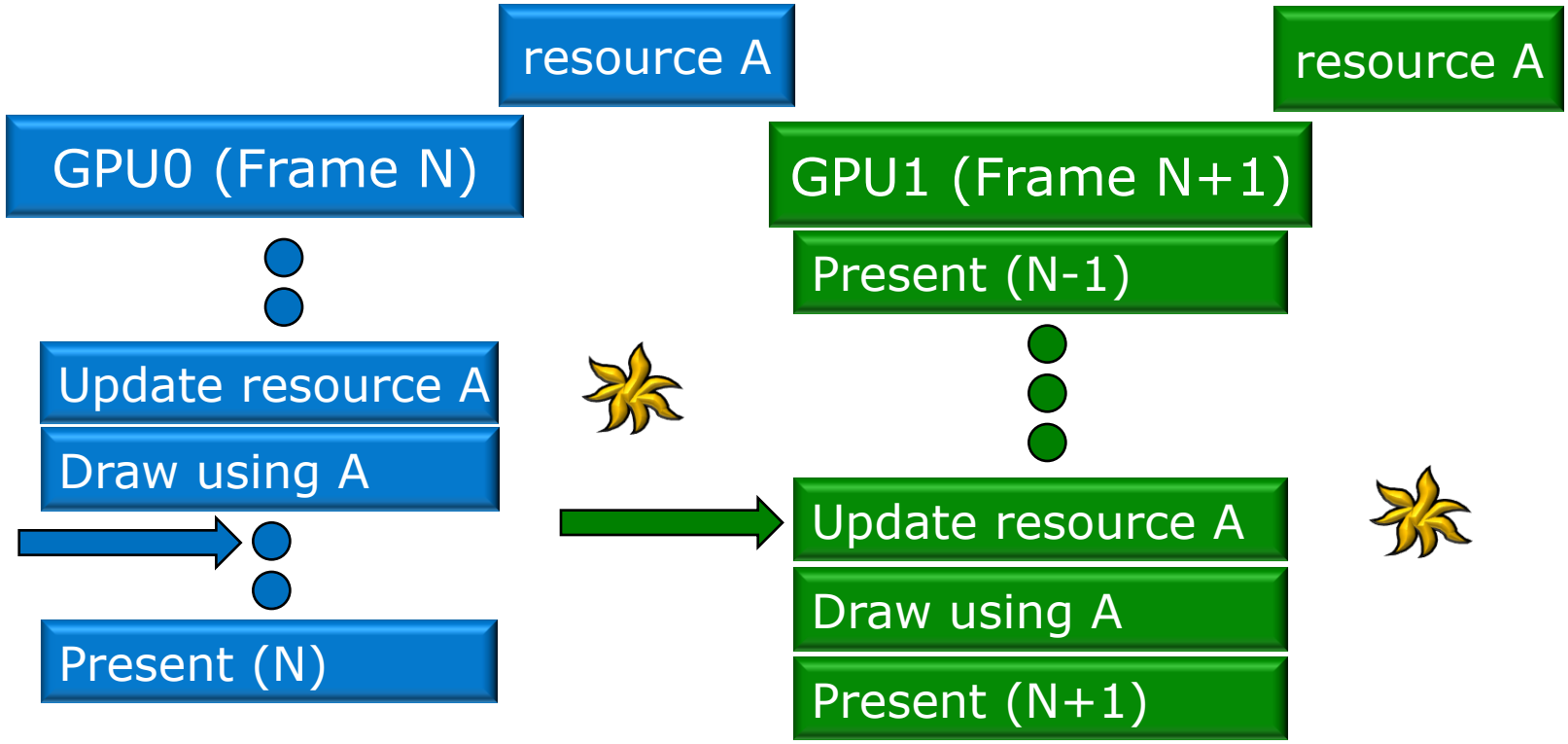


Solution: Resources that Change Every Frame

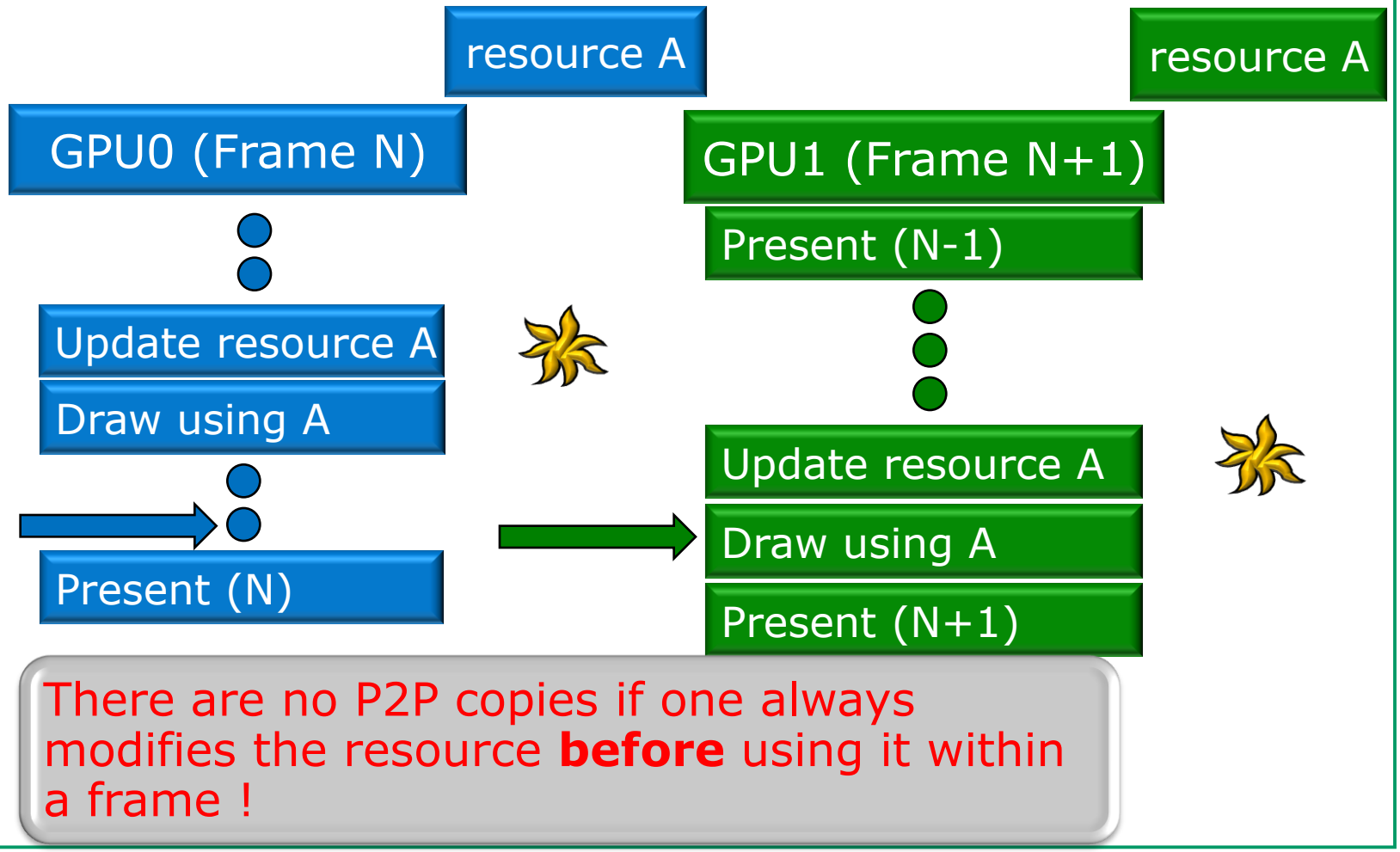


Solution: Resources that Change Every Frame

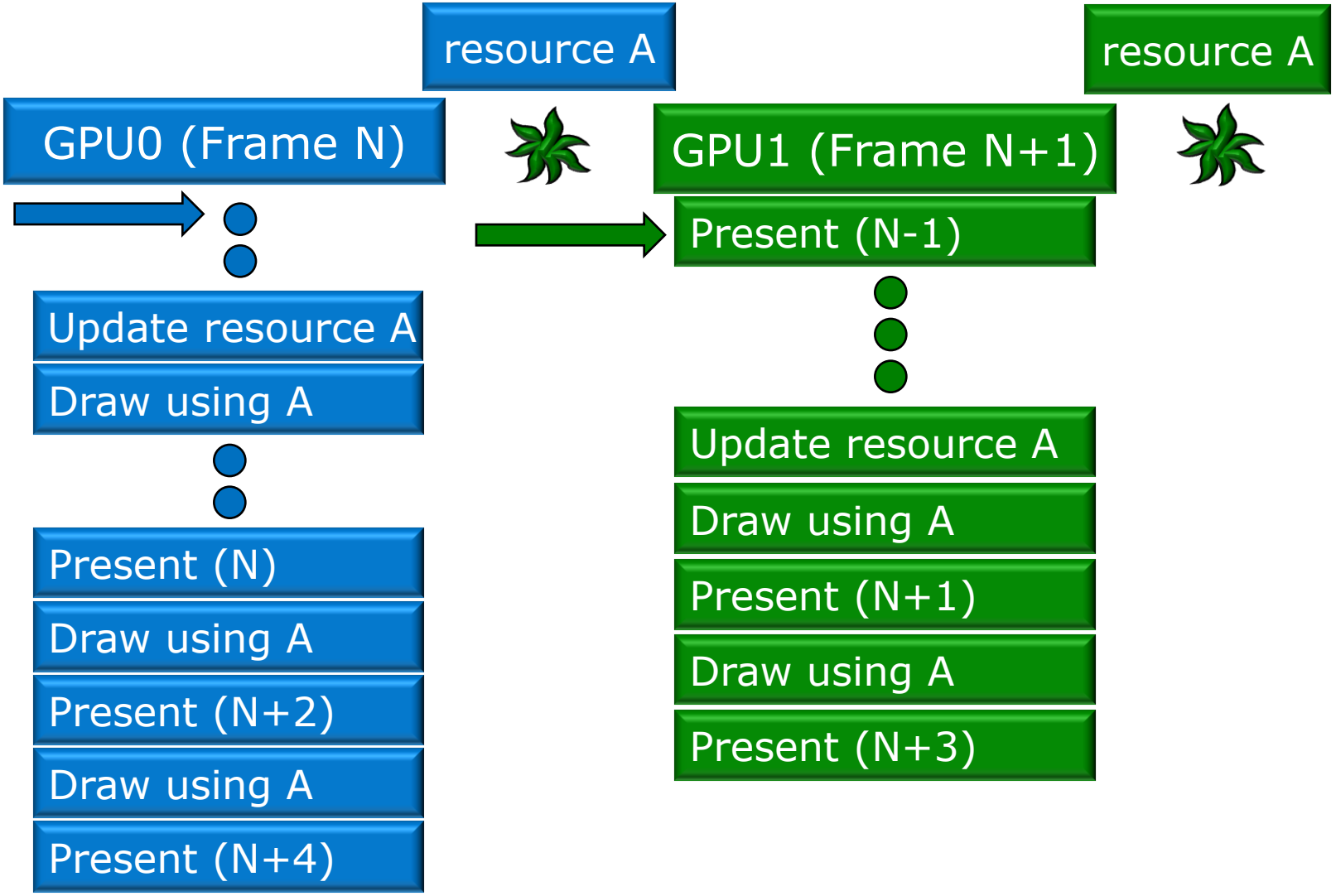
AMD
Smarter Choice



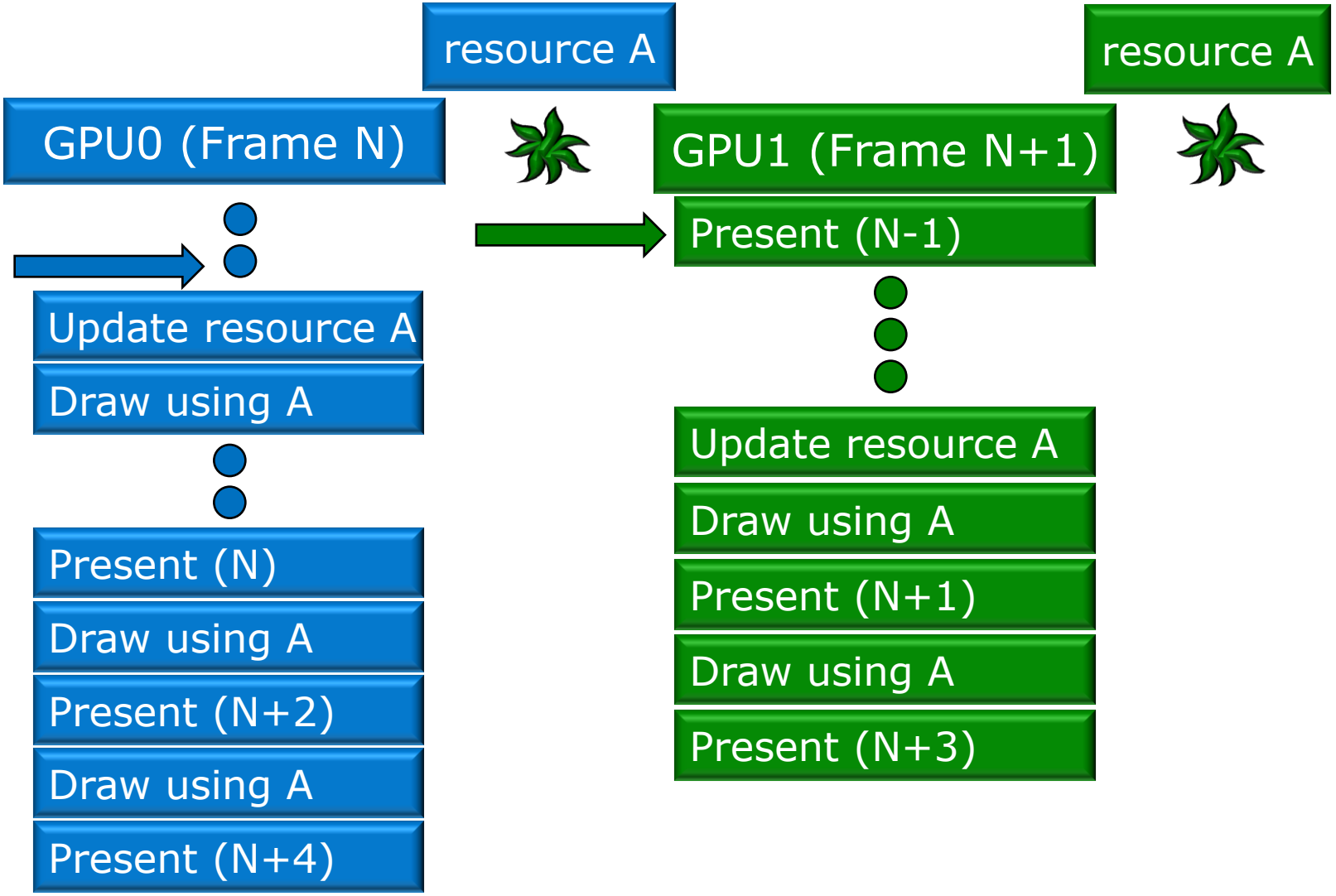
Solution: Resources that Change Every Frame



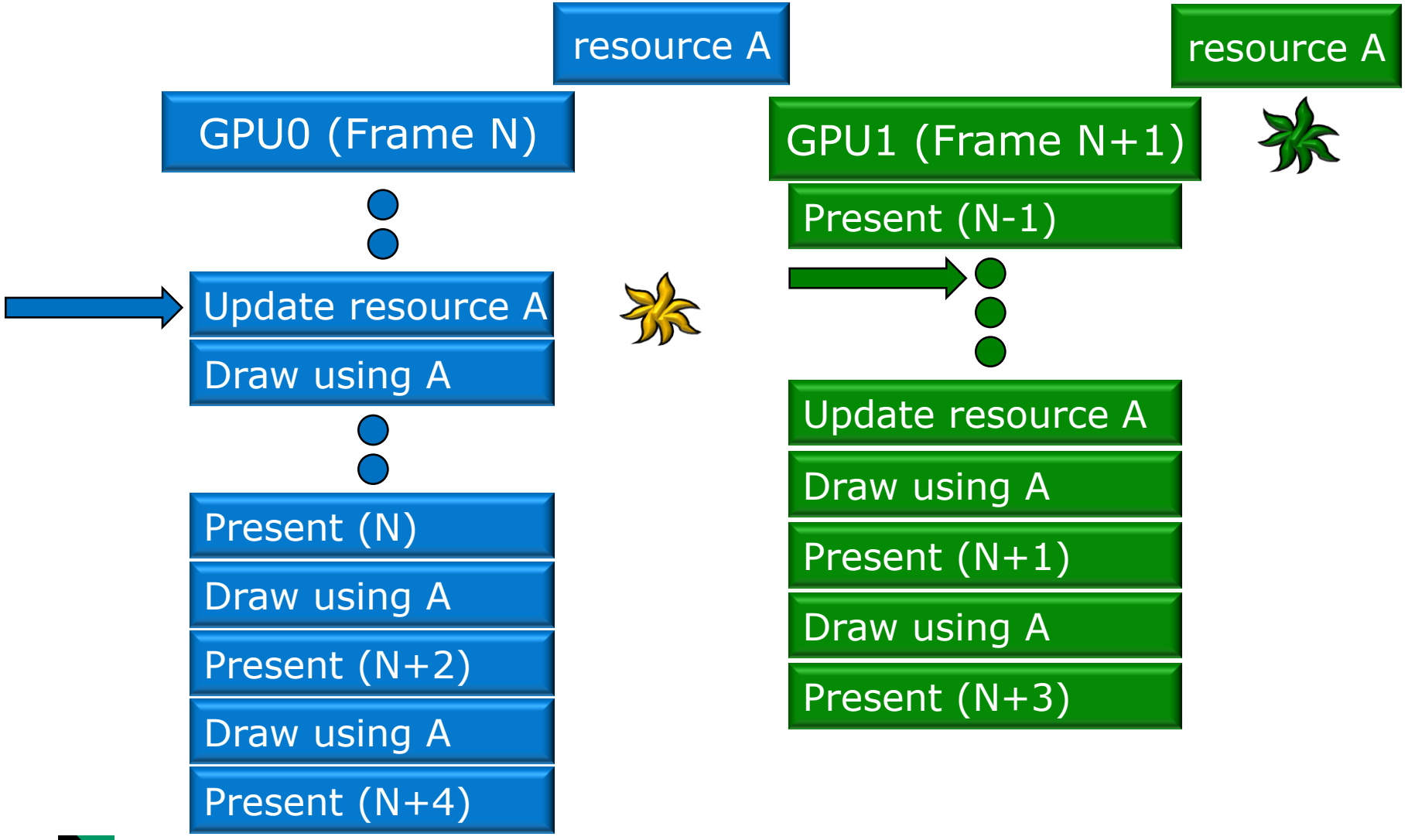
Solution: Resources that Change Every Few Frames



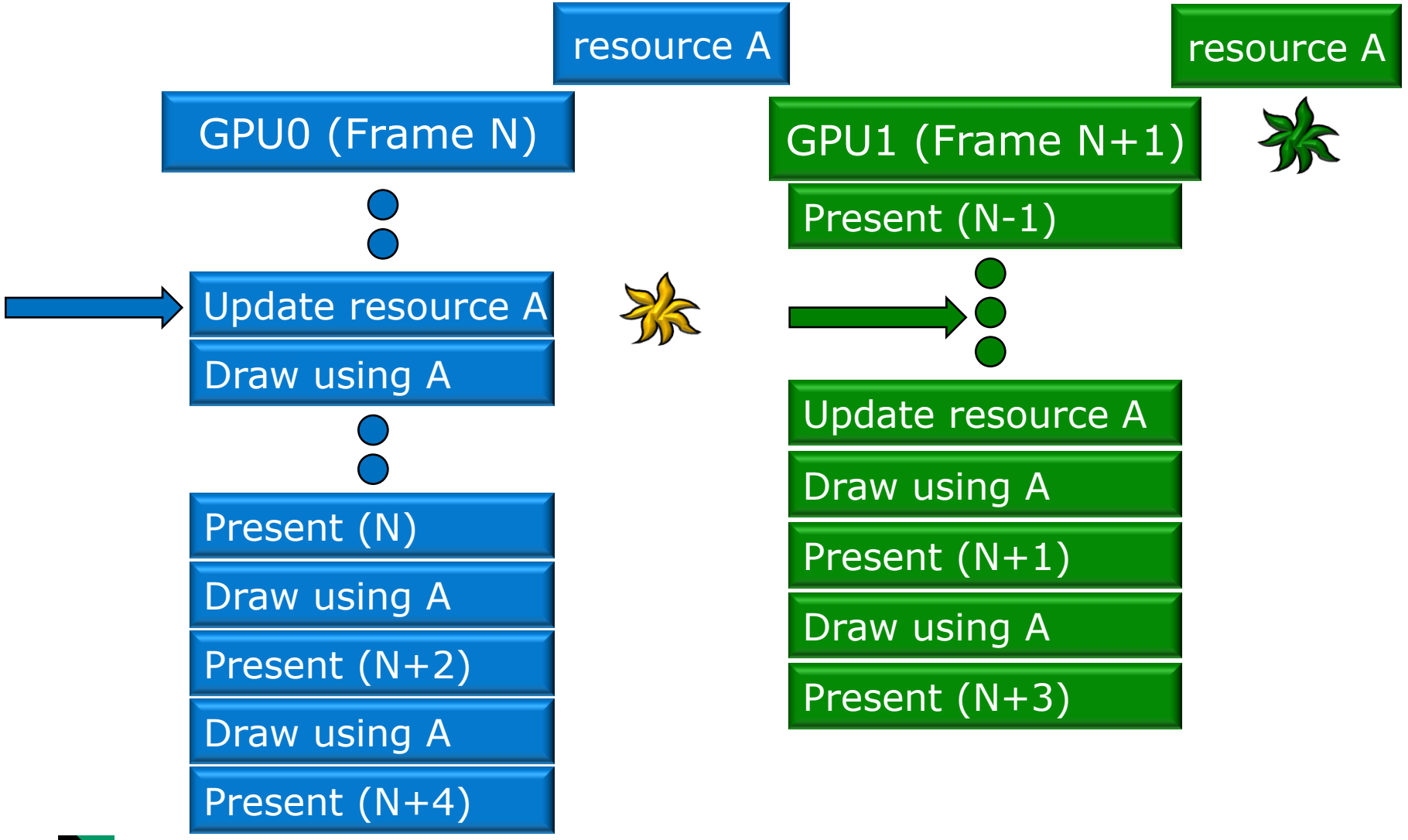
Solution: Resources that Change Every Few Frames



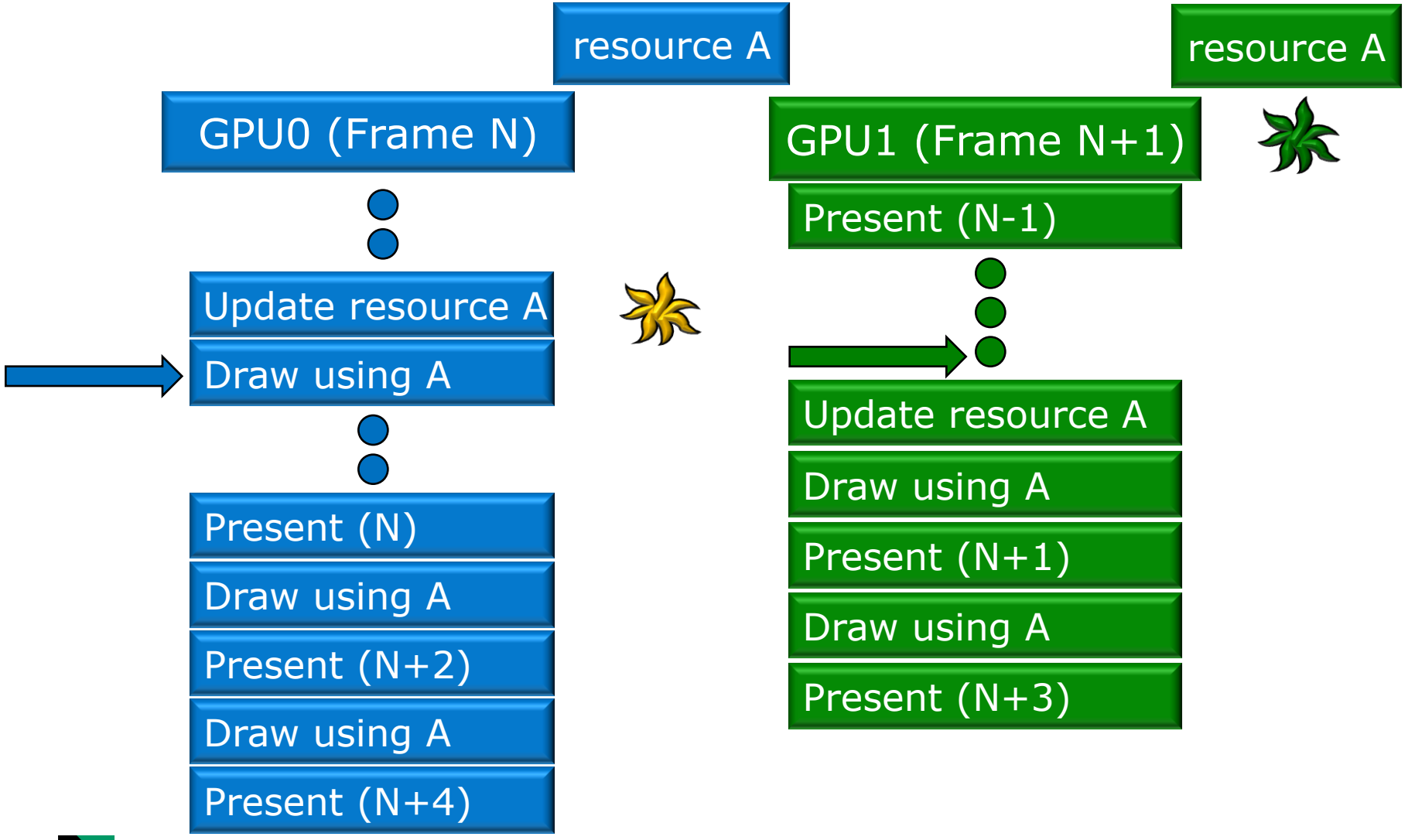
Solution: Resources that Change Every Few Frames



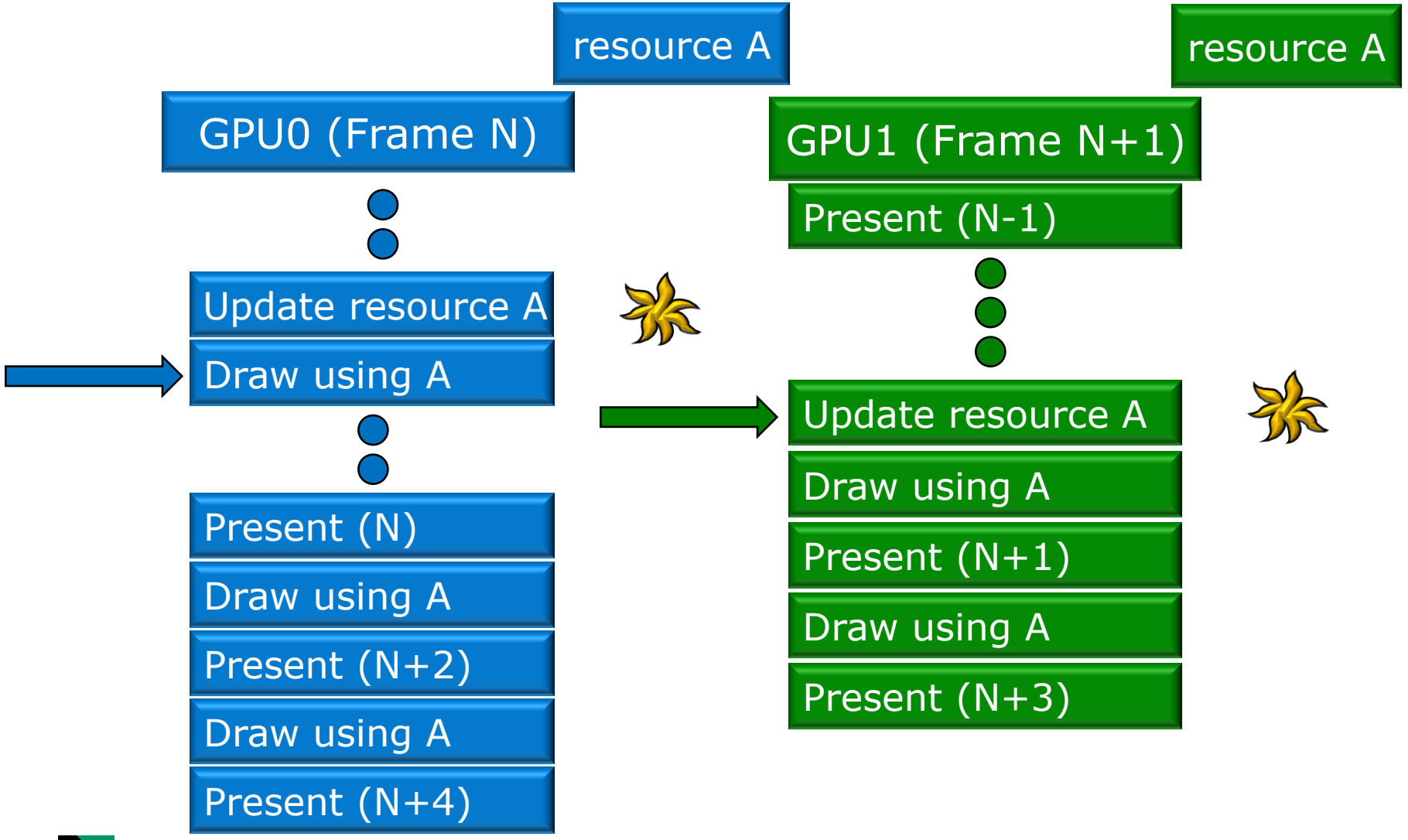
Solution: Resources that Change Every Few Frames



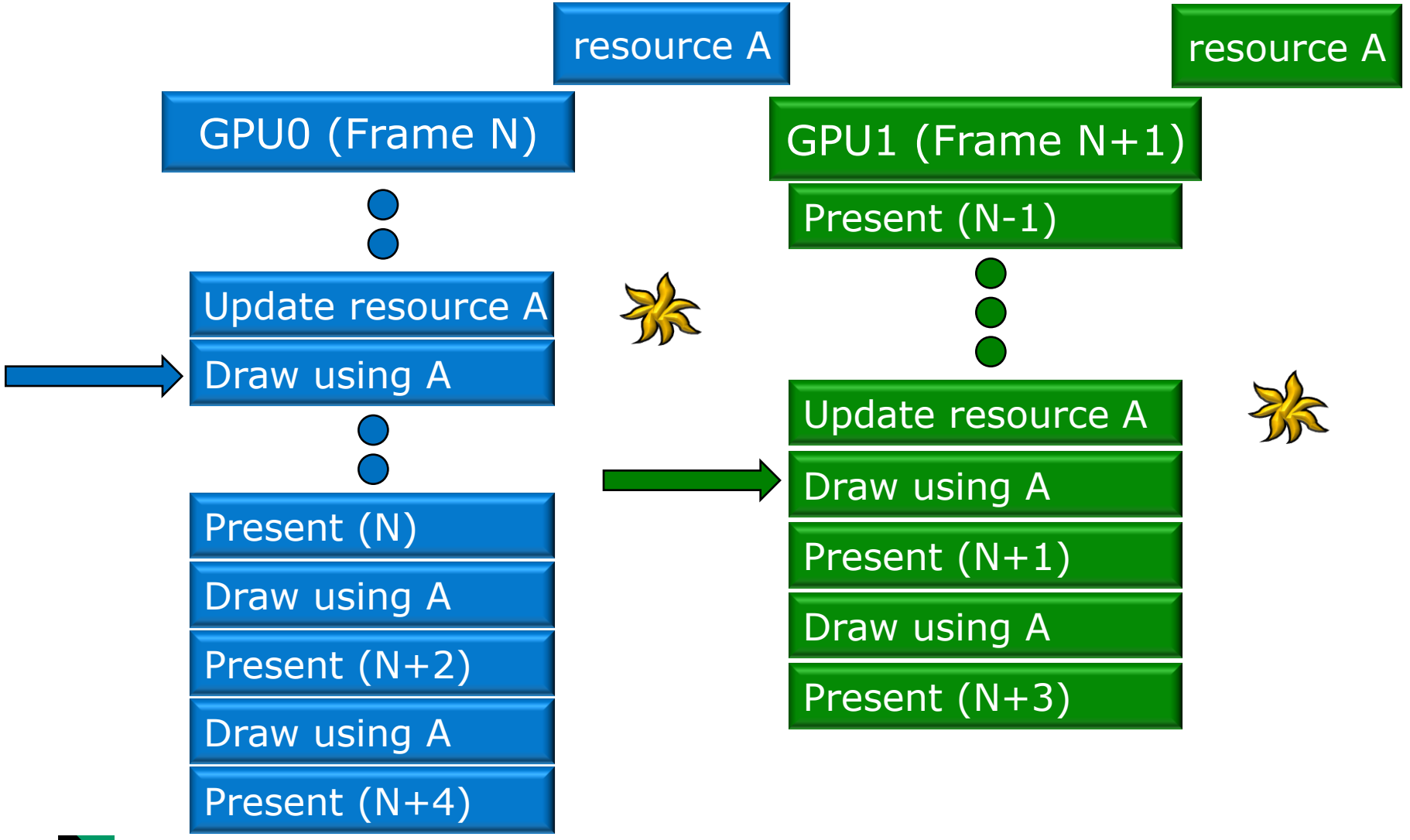
Solution: Resources that Change Every Few Frames



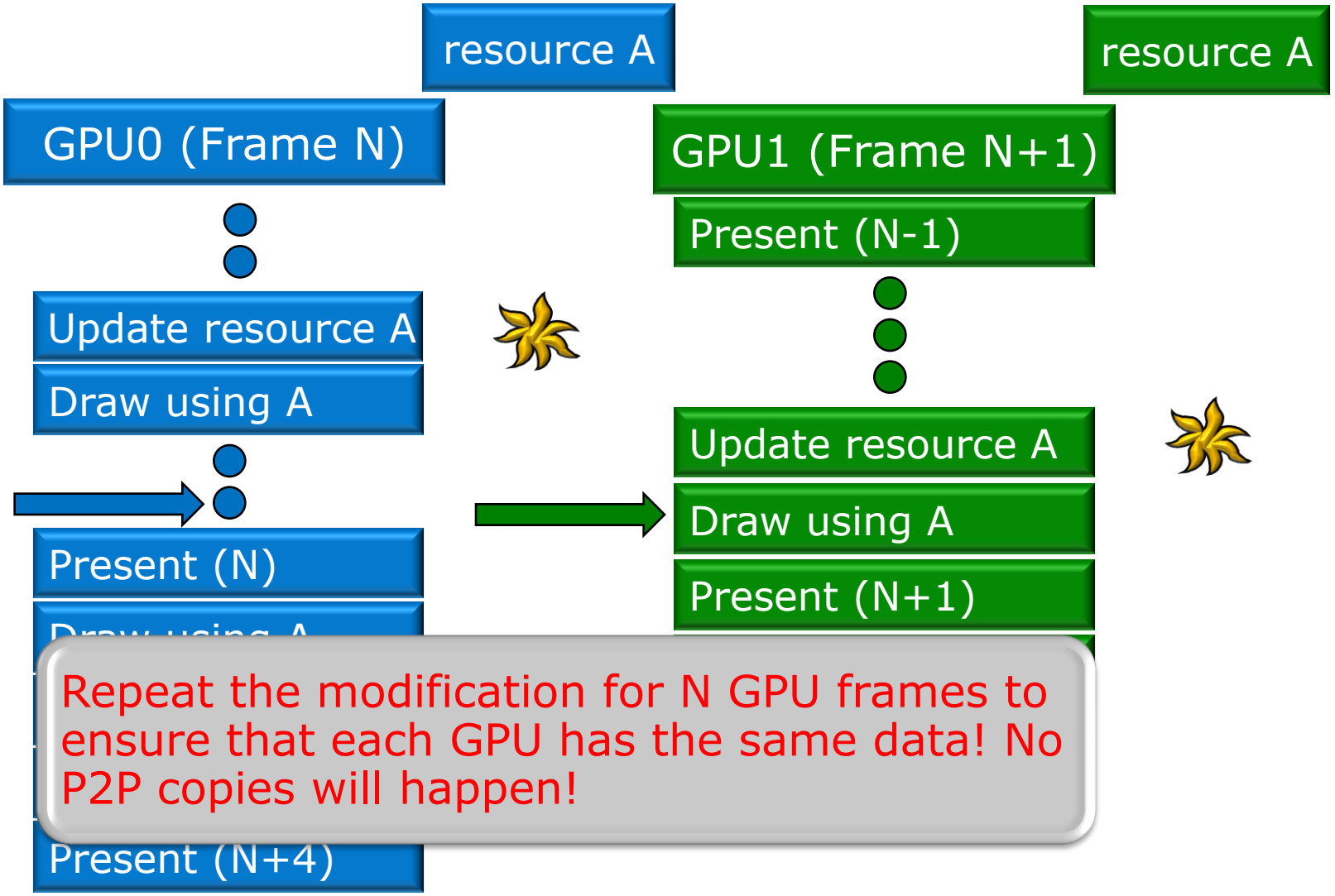
Solution: Resources that Change Every Few Frames



Solution: Resources that Change Every Few Frames



Solution: Resources that Change Every Few Frames

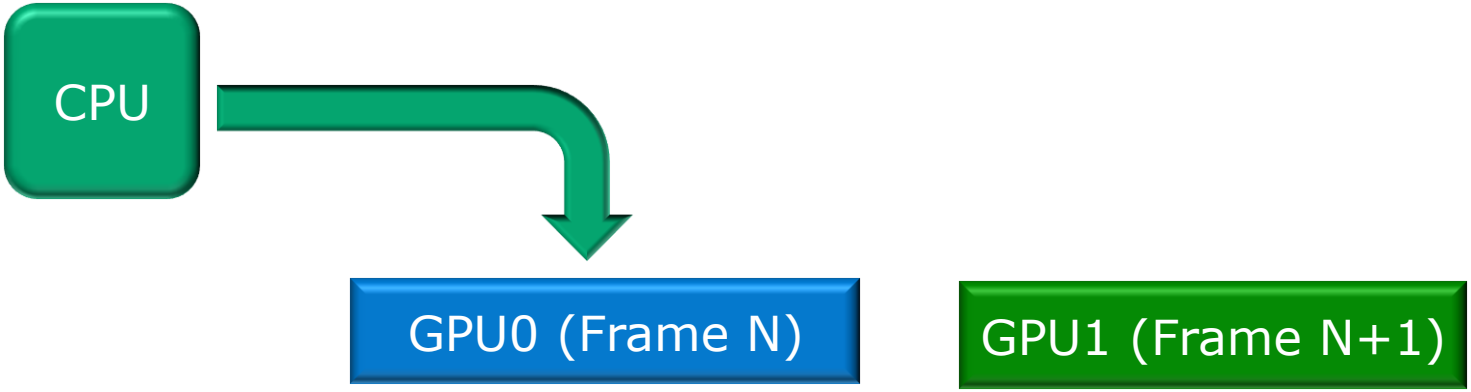


Pitfalls: In DX10 there are Other Ways to Update Resources...

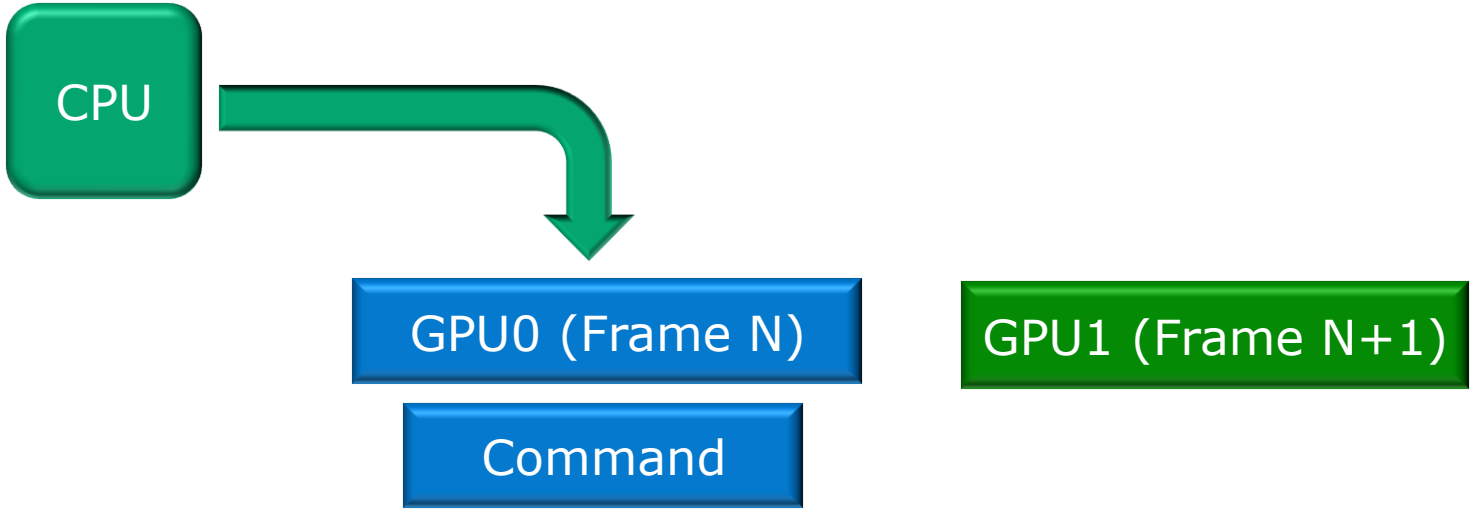
AMD
Smarter Choice

- Drawing to vertex/index buffers
- Stream Out
- CopyResource() calls
- CopySubresourceRegion() calls
- GenerateMips() calls
- ResolveSubresource() calls

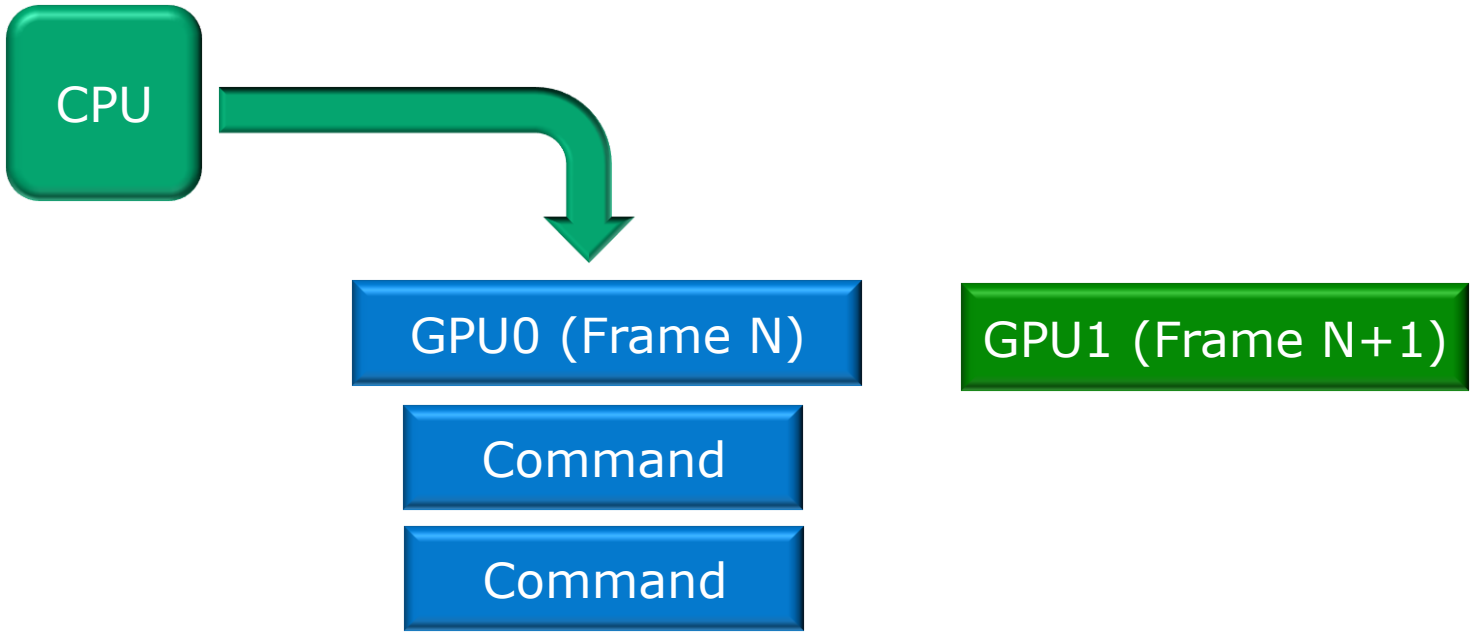
Pitfall: Waiting on Queries



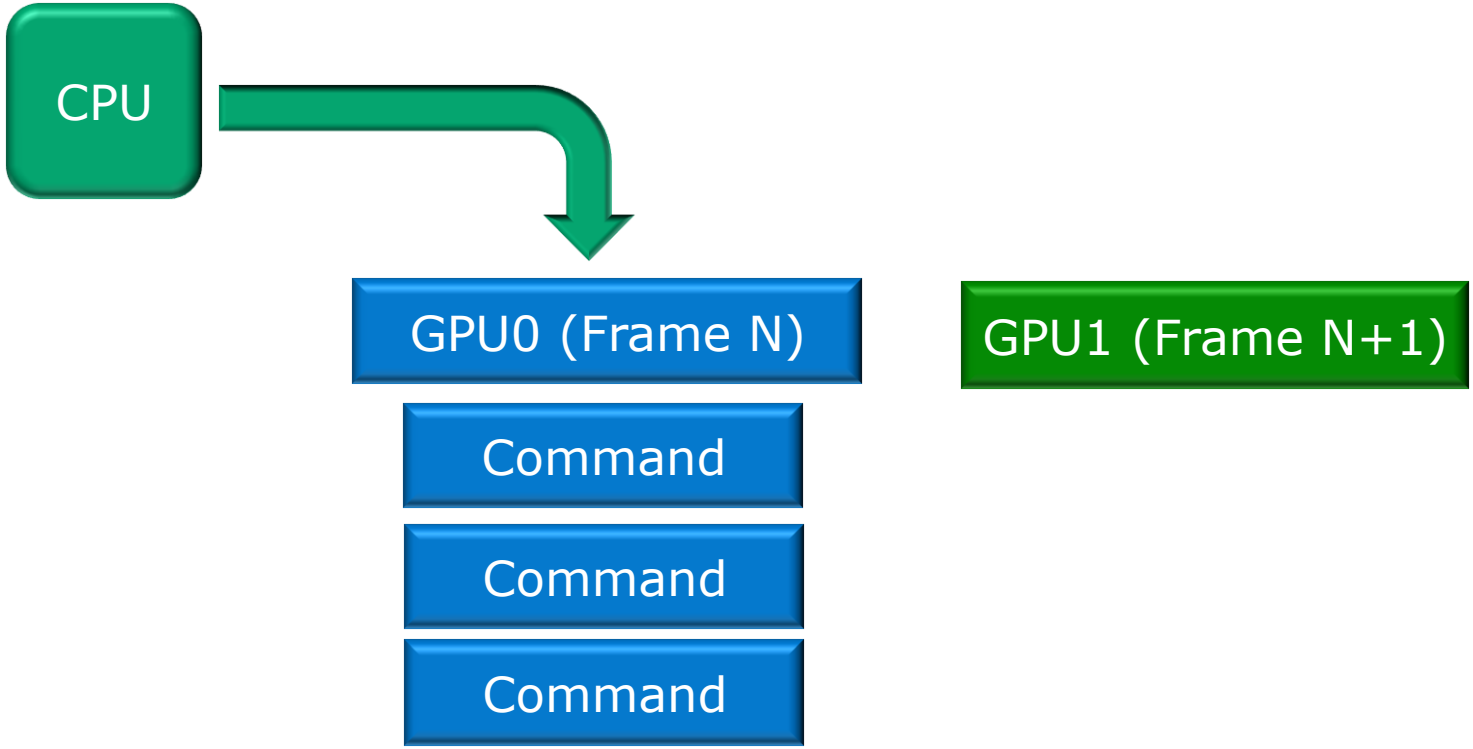
Pitfall: Waiting on Queries



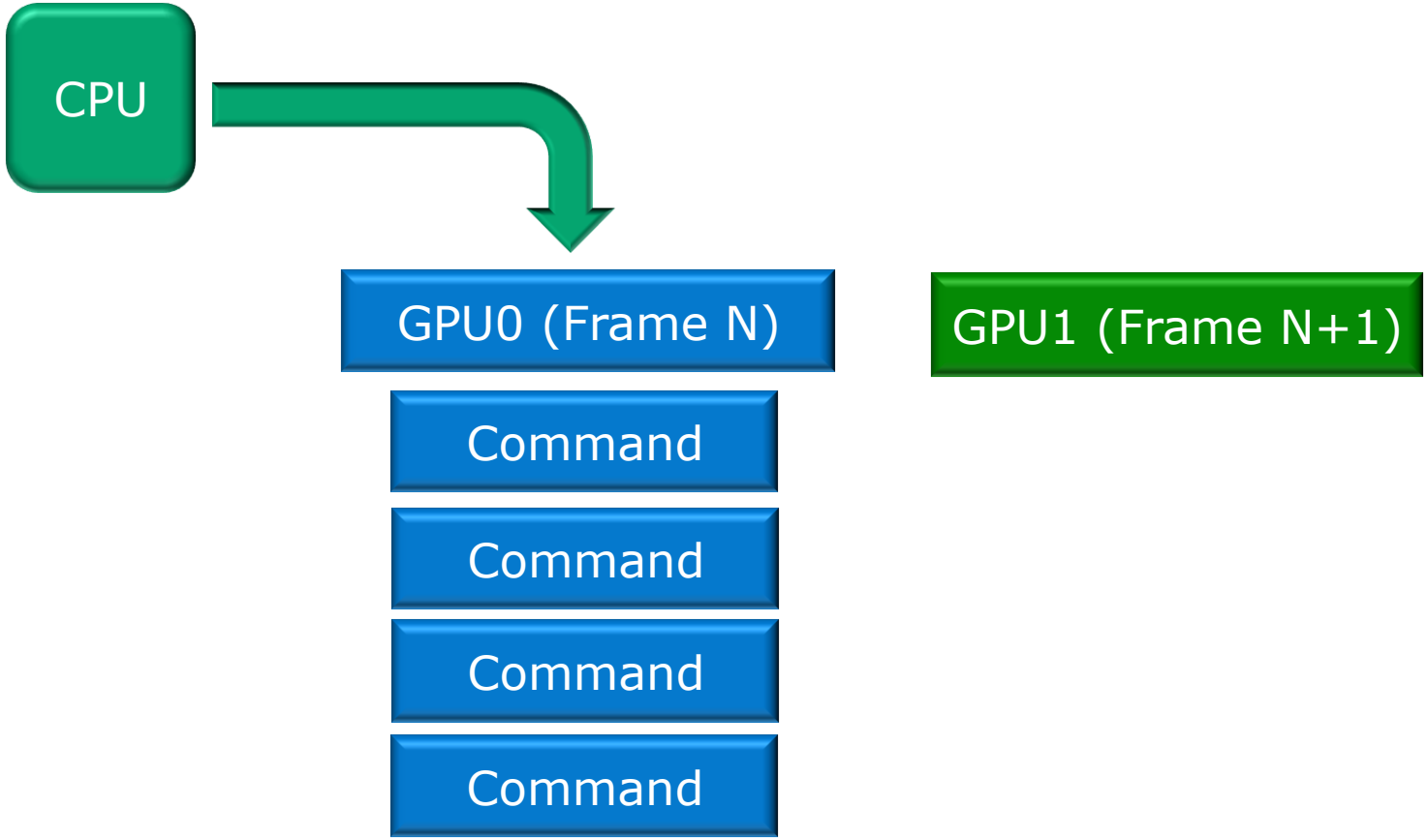
Pitfall: Waiting on Queries



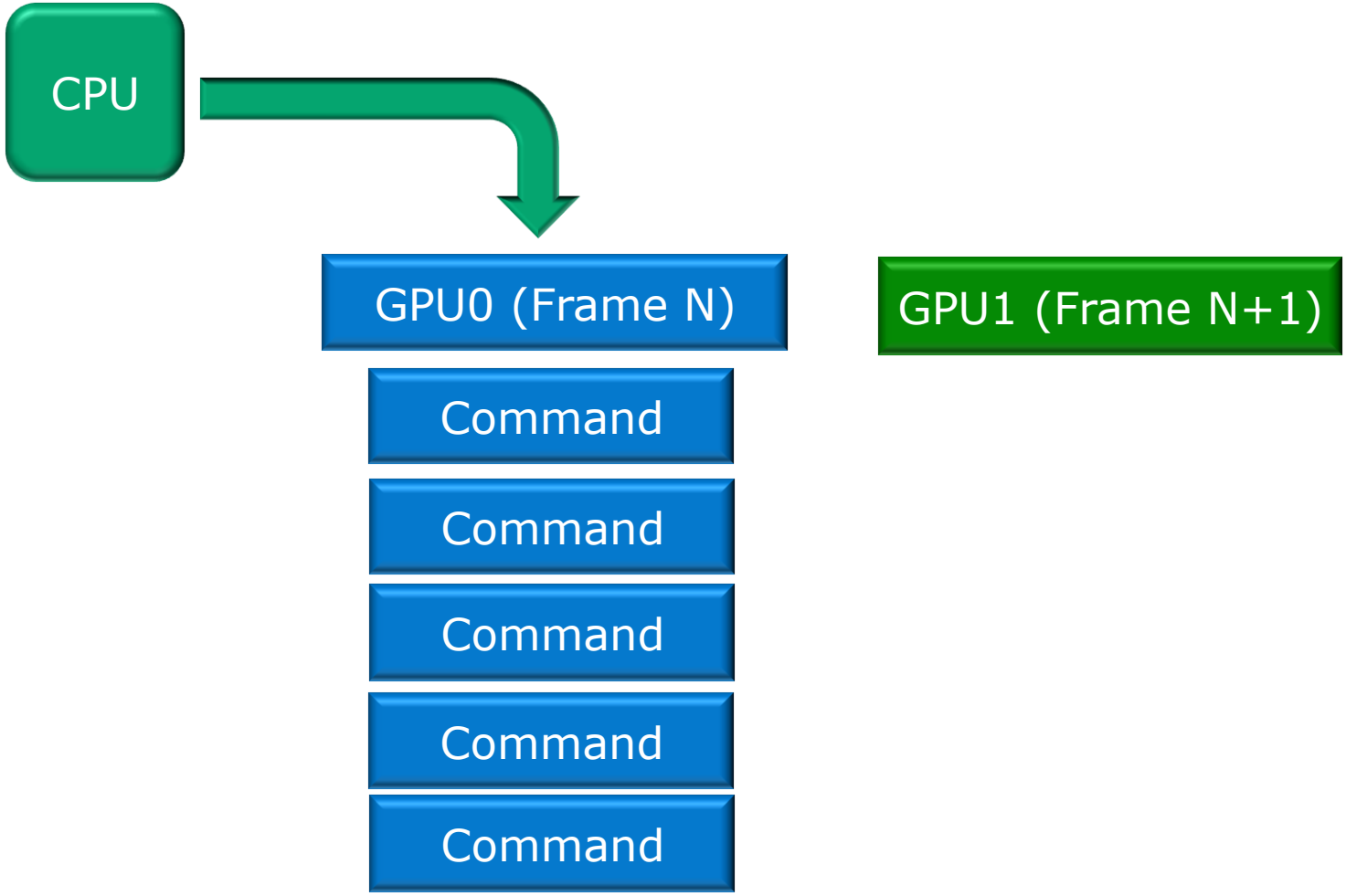
Pitfall: Waiting on Queries



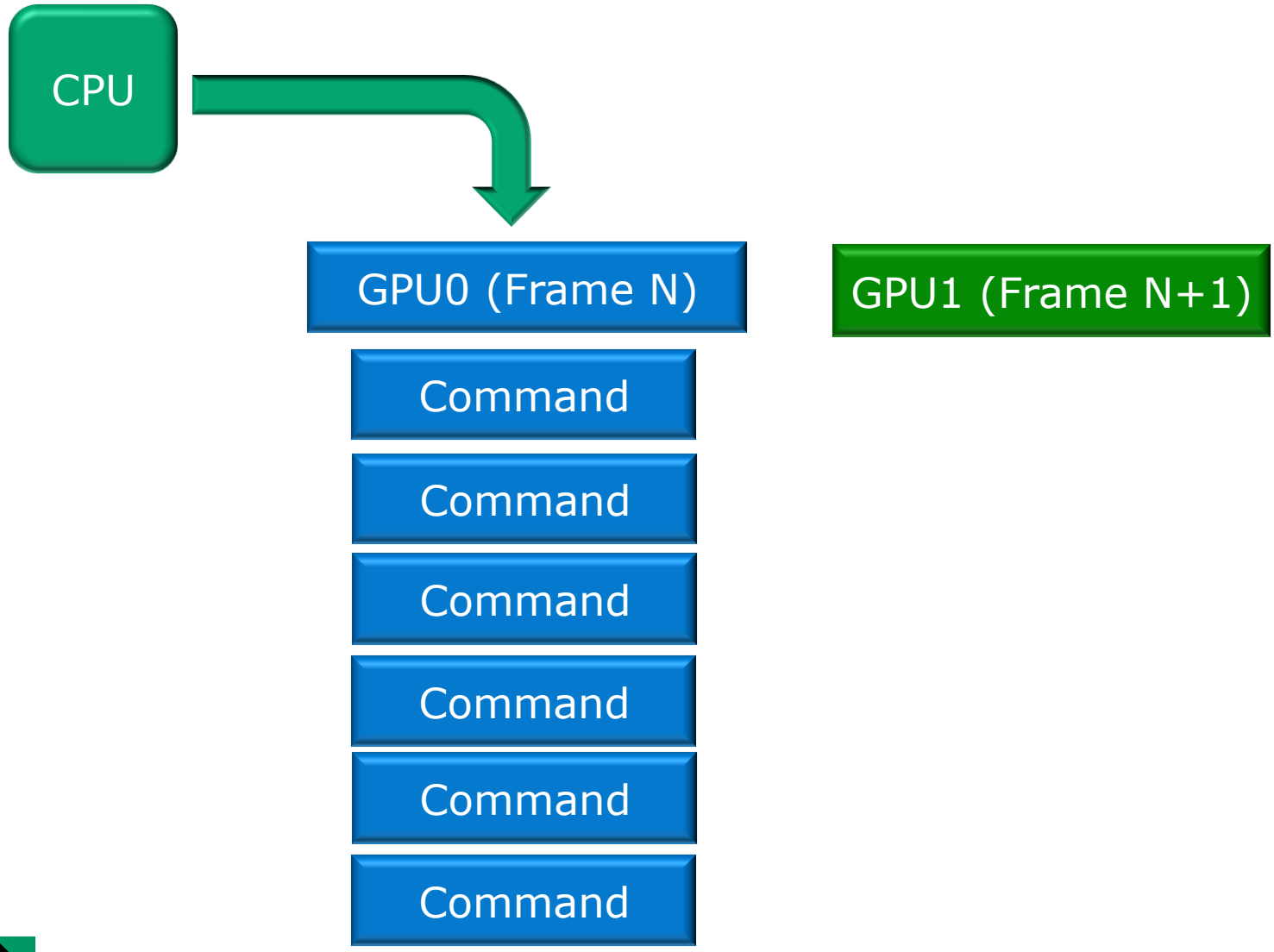
Pitfall: Waiting on Queries



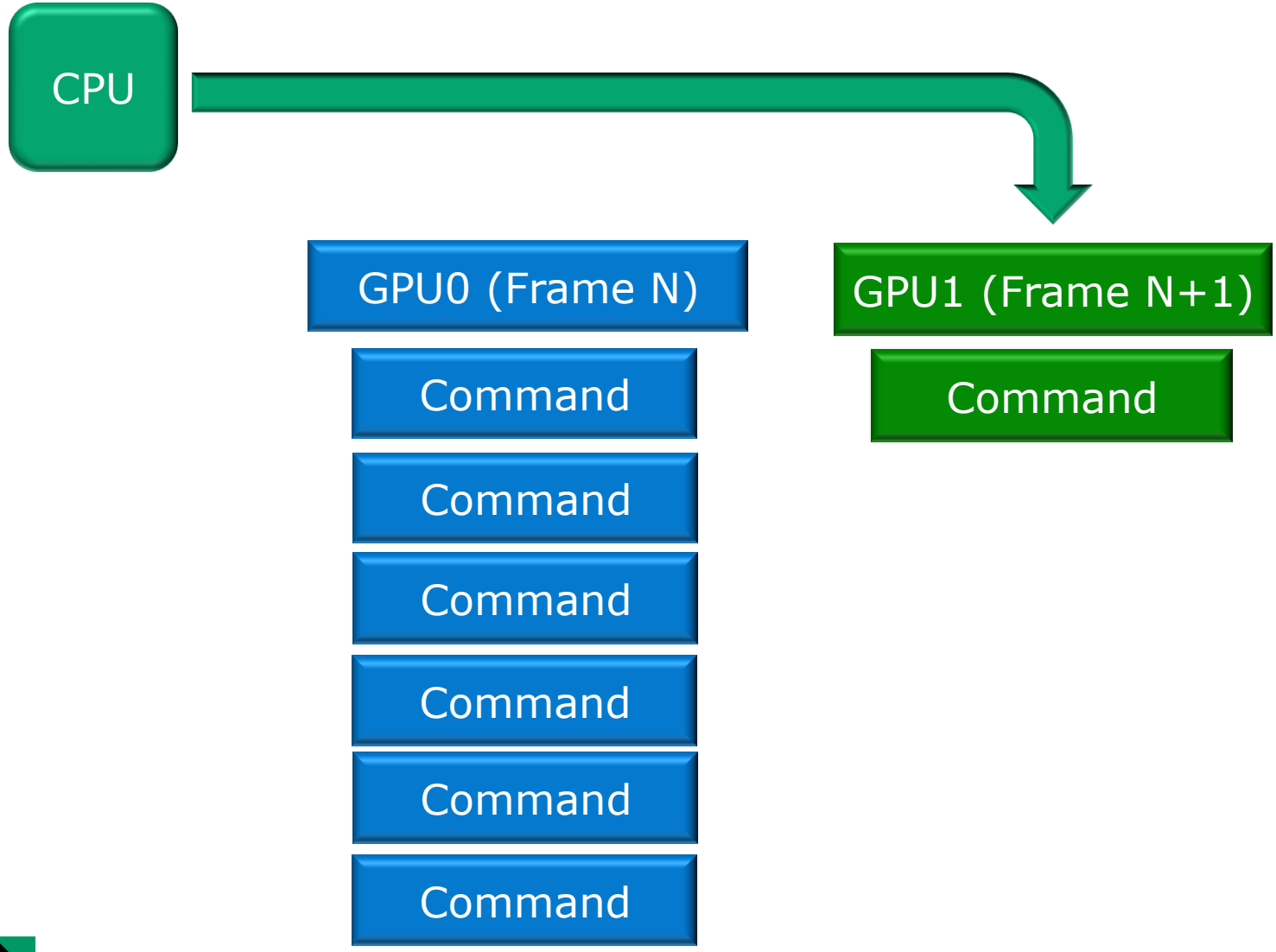
Pitfall: Waiting on Queries



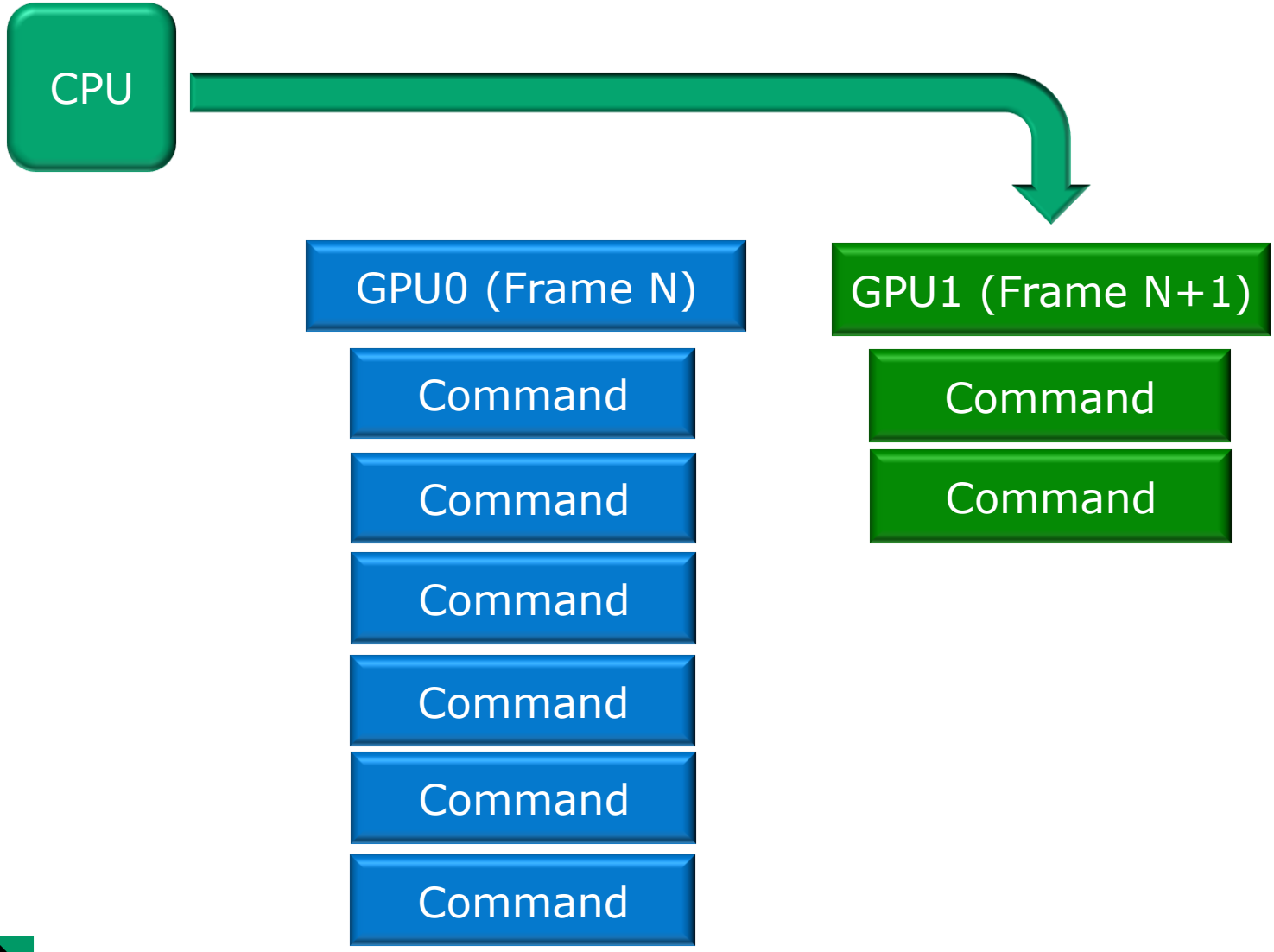
Pitfall: Waiting on Queries



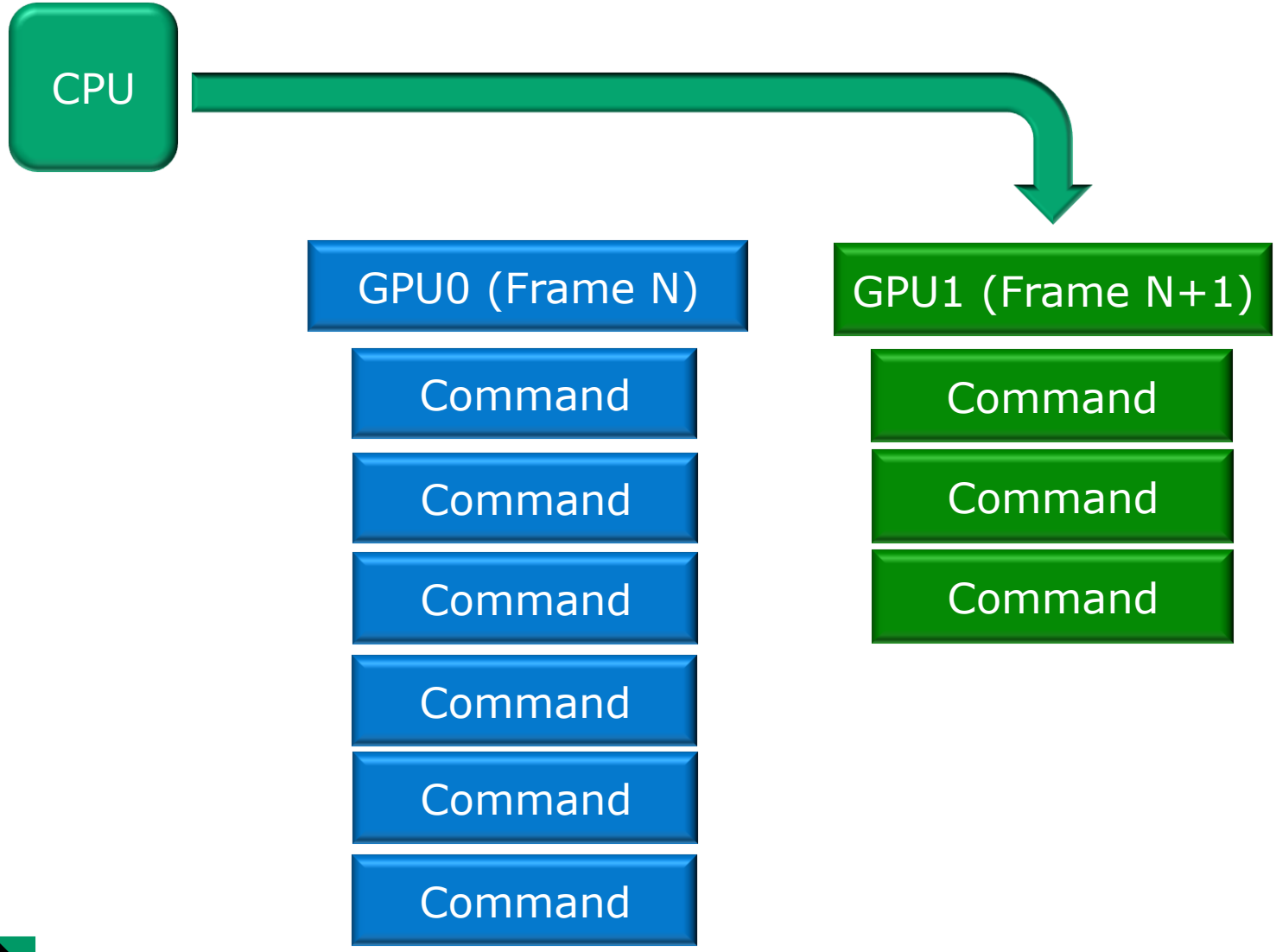
Pitfall: Waiting on Queries



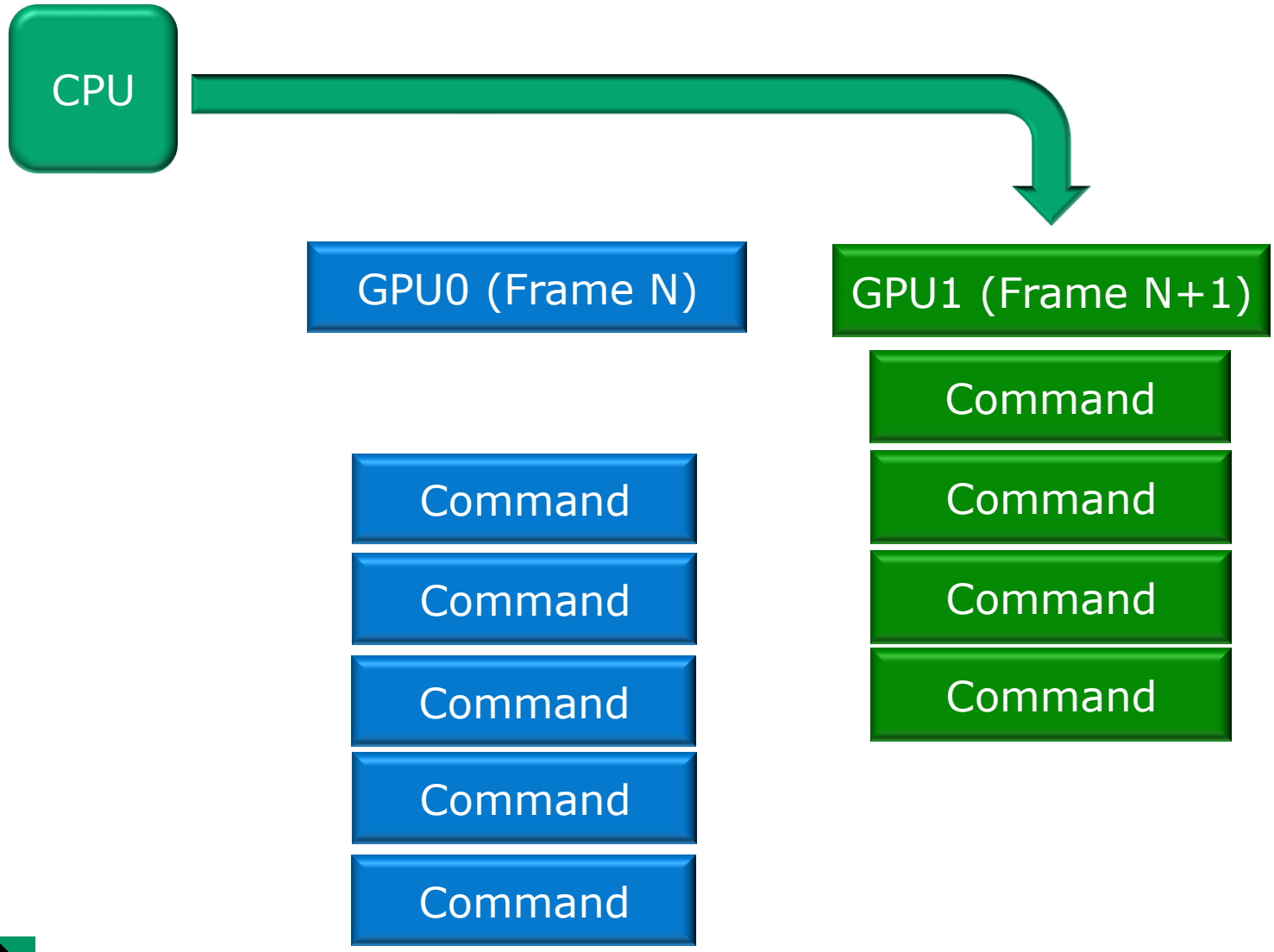
Pitfall: Waiting on Queries



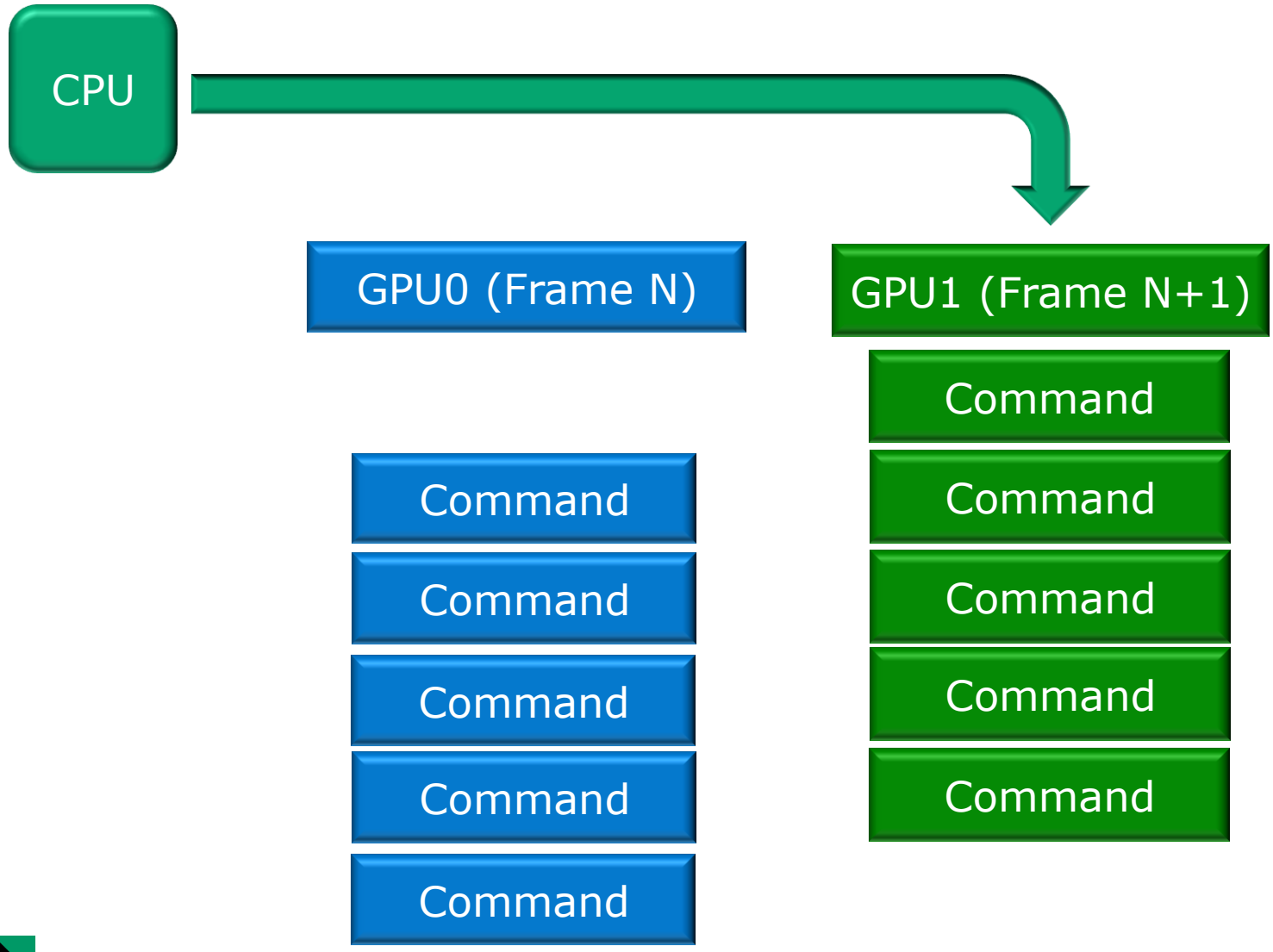
Pitfall: Waiting on Queries



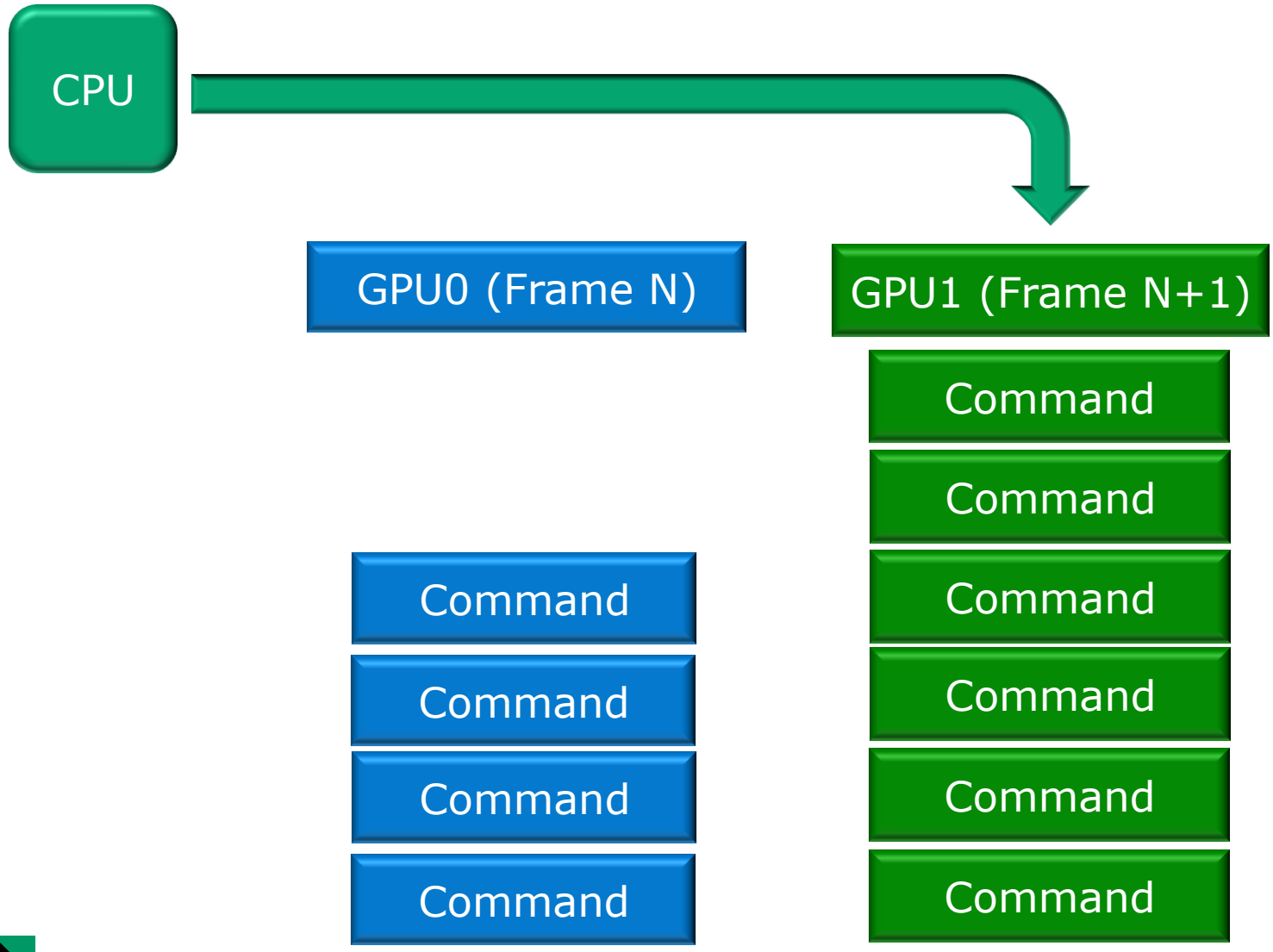
Pitfall: Waiting on Queries



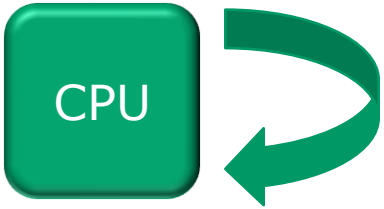
Pitfall: Waiting on Queries



Pitfall: Waiting on Queries



Pitfall: Waiting on Queries



Waiting for Query Result!!!

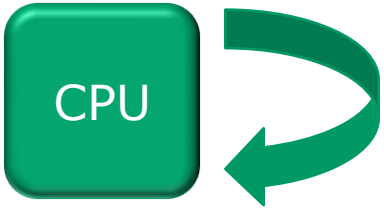
GPU0 (Frame N)

GPU1 (Frame N+1)

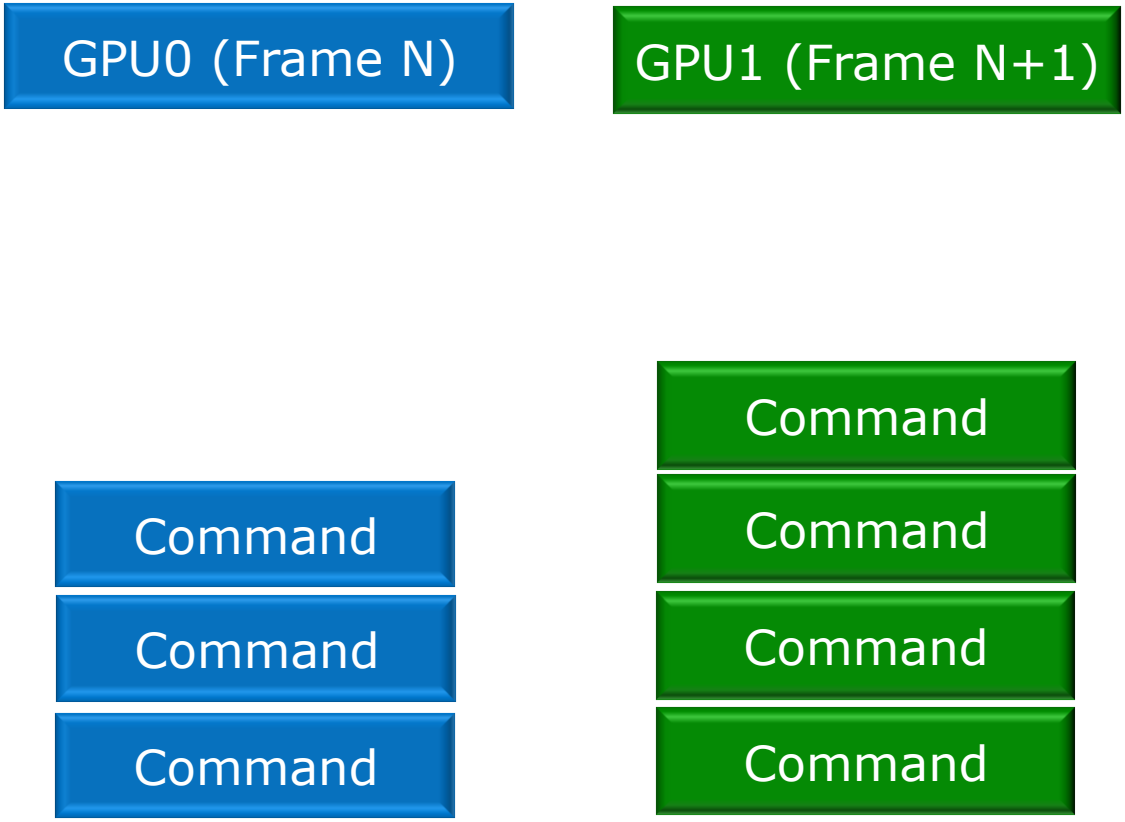
Command
Command
Command
Command

Command
Command
Command
Command
Command

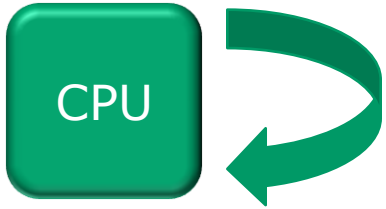
Pitfall: Waiting on Queries



Waiting for Query Result!!!



Pitfall: Waiting on Queries



Waiting for Query Result!!!

GPU0 (Frame N)

GPU1 (Frame N+1)

Command

Command

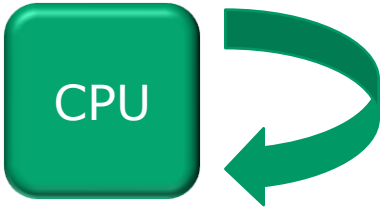
Command

Command

Command

Command

Pitfall: Waiting on Queries



Waiting for Query Result!!!

GPU0 (Frame N)

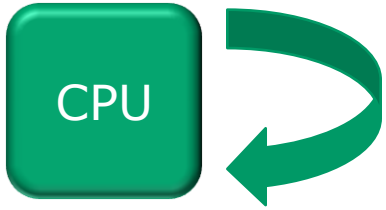
GPU1 (Frame N+1)

Command

Command

Command

Pitfall: Waiting on Queries



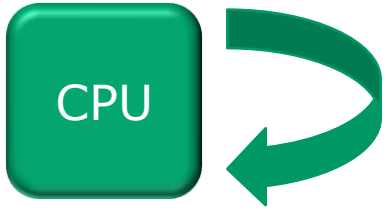
Waiting for Query Result!!!

GPU0 (Frame N)

GPU1 (Frame N+1)

Command

Pitfall: Waiting on Queries

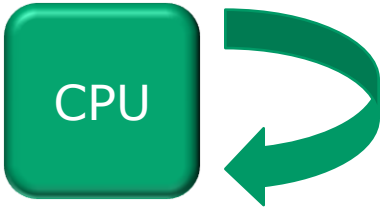


Waiting for Query Result!!!

GPU0 (Frame N)

GPU1 (Frame N+1)

Pitfall: Waiting on Queries



Waiting for Query Result!!!

GPU0 (Frame N)

GPU1 (Frame N+1)

Waiting starves GPU queues

Waiting limits parallelism

Waiting => CPU limitation

Solution: Queries

AMD
Smarter Choice

- Avoid using queries wherever possible
- Make sure that a matched pair of BeginQuery / EndQuery calls occur within the same frame
- Avoid waiting on query results
- For queries issued every frame
  - Create additional query objects for each GPU
  - Cycle through them
- Pickup the result of a query at least N-GPU frames after it was issued
- For occlusion queries consider a CPU based approach

Pitfall: CPU Access to a Renderable Resource

AMD
Smarter Choice

- When the CPU locks a renderable resource it must wait for all GPUs to finish using the resource before acquiring the pointer
- All GPUs now have to wait until the CPU unlocks the resource pointer
- After the unlock the driver has to update the resource on each GPU via P2P copies
- Just don‘t do this – it destroys performance even on a single GPU setup, and is catastrophic for MGPUs

Solutions: Locks / Maps

AMD
Smarter Choice

- In DX10 stream to and copy from STAGING textures
- In DX9 StretchRect() is always better than Lock()
- At resource creation time use the appropriate flags from:
  - D3D10_USAGE
  - D3D10_CPU_ACCESS_FLAG
- In DX9 never lock static Vertex/Index Buffers because it will cause P2P copies

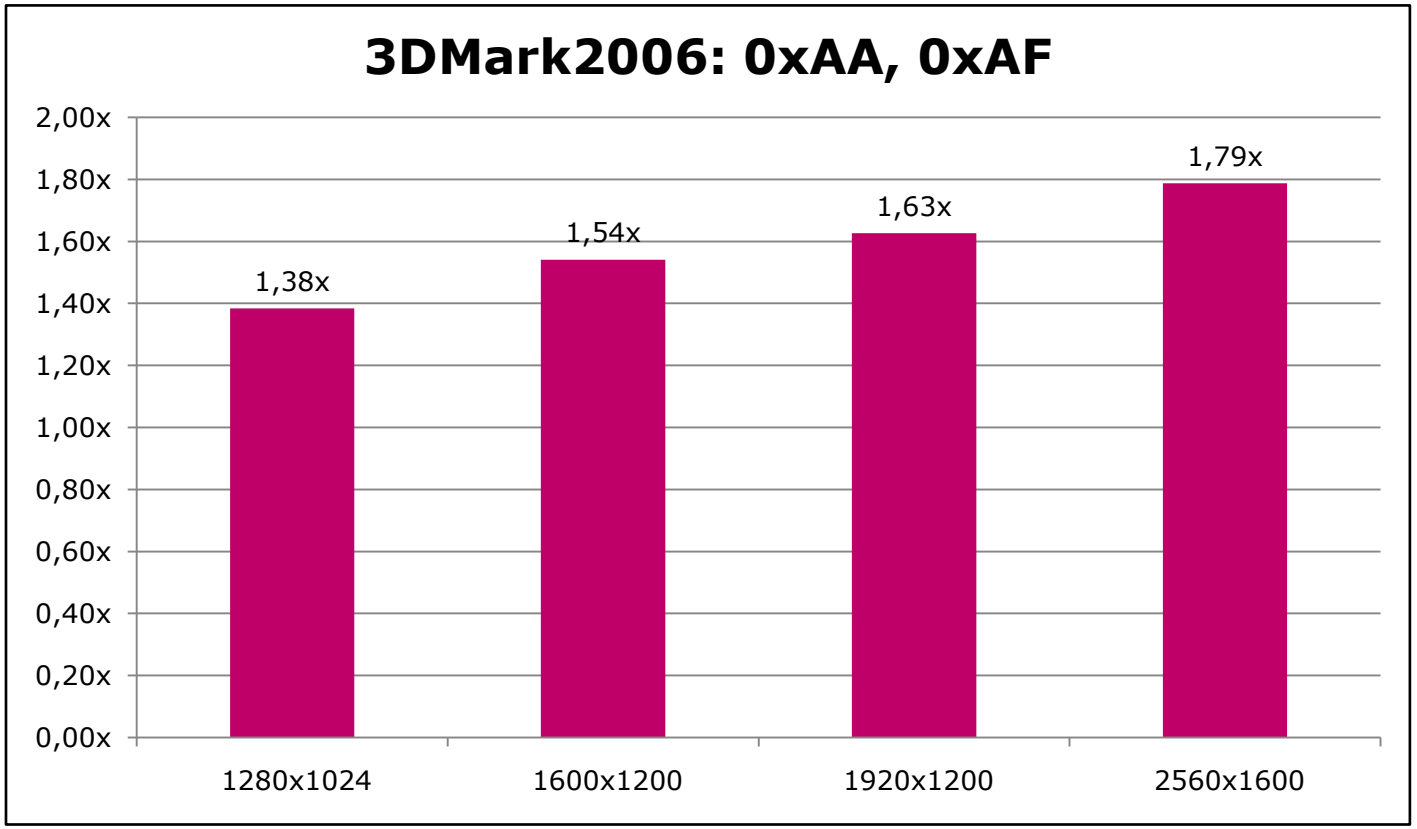
Concluding Pitfalls & Solutions

AMD
Smarter Choice

- Drivers take a conservative approach
  - Performs checks on resource synchronization
  - P2P copy if necessary
- You know the application best
  - Determine if a P2P copy is necessary
  - Talk to us about a profile
- In DX10 never use a resource shared between devices
  - No way to detect updates to resources by other applications

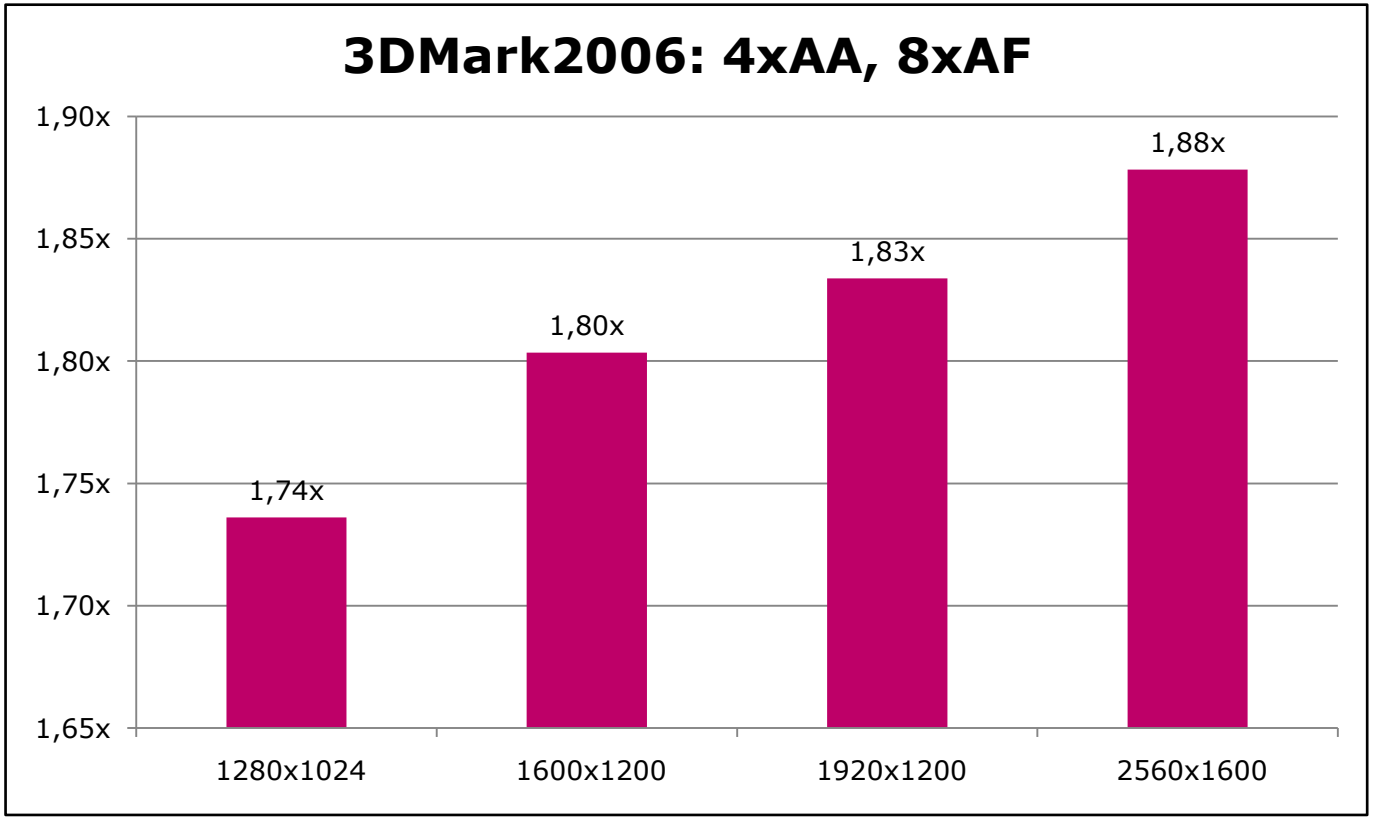
Performance Gains

3DMark2006



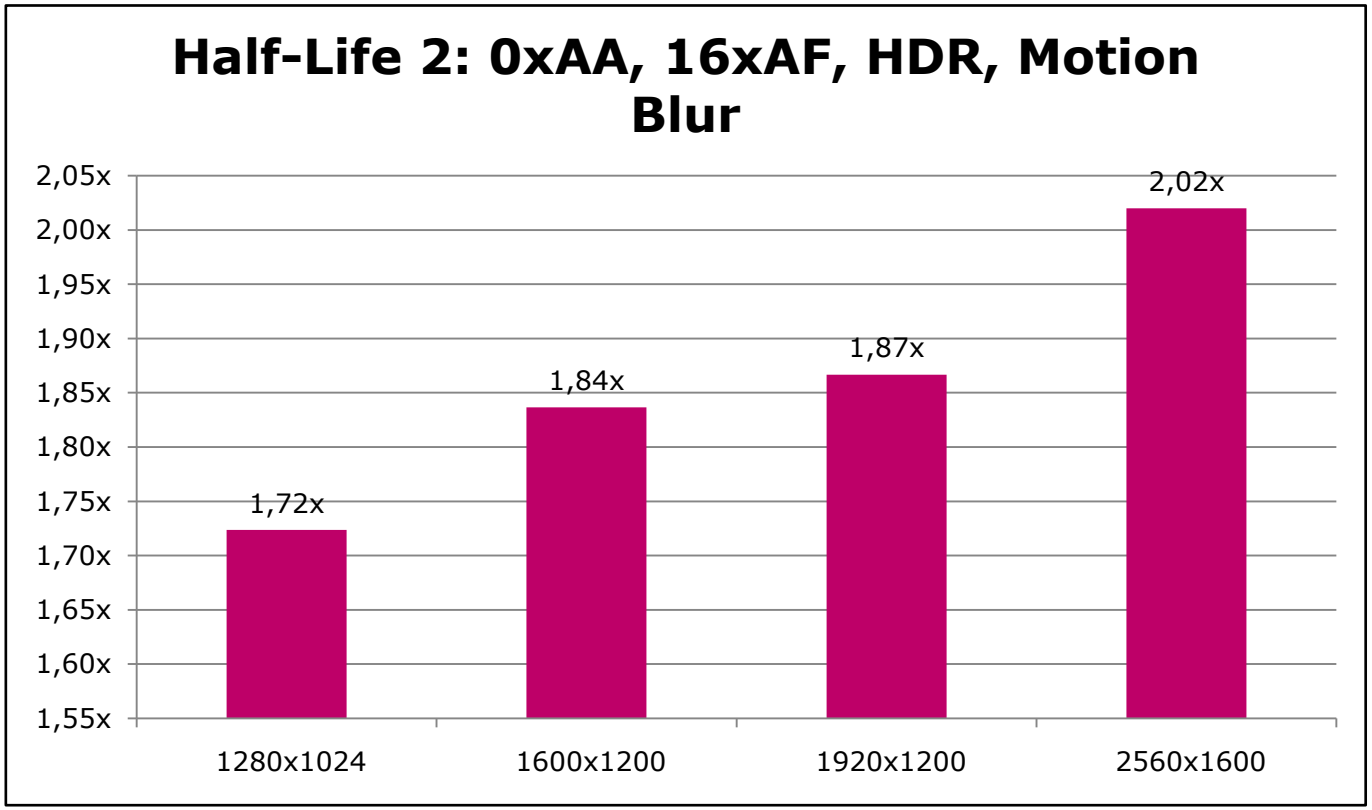
* HD3870 vs HD3870 X2 (QX9650 (3.0/1333), ASUS X38, 2GB DDR2-800, Windows VISTA64)

3DMark2006



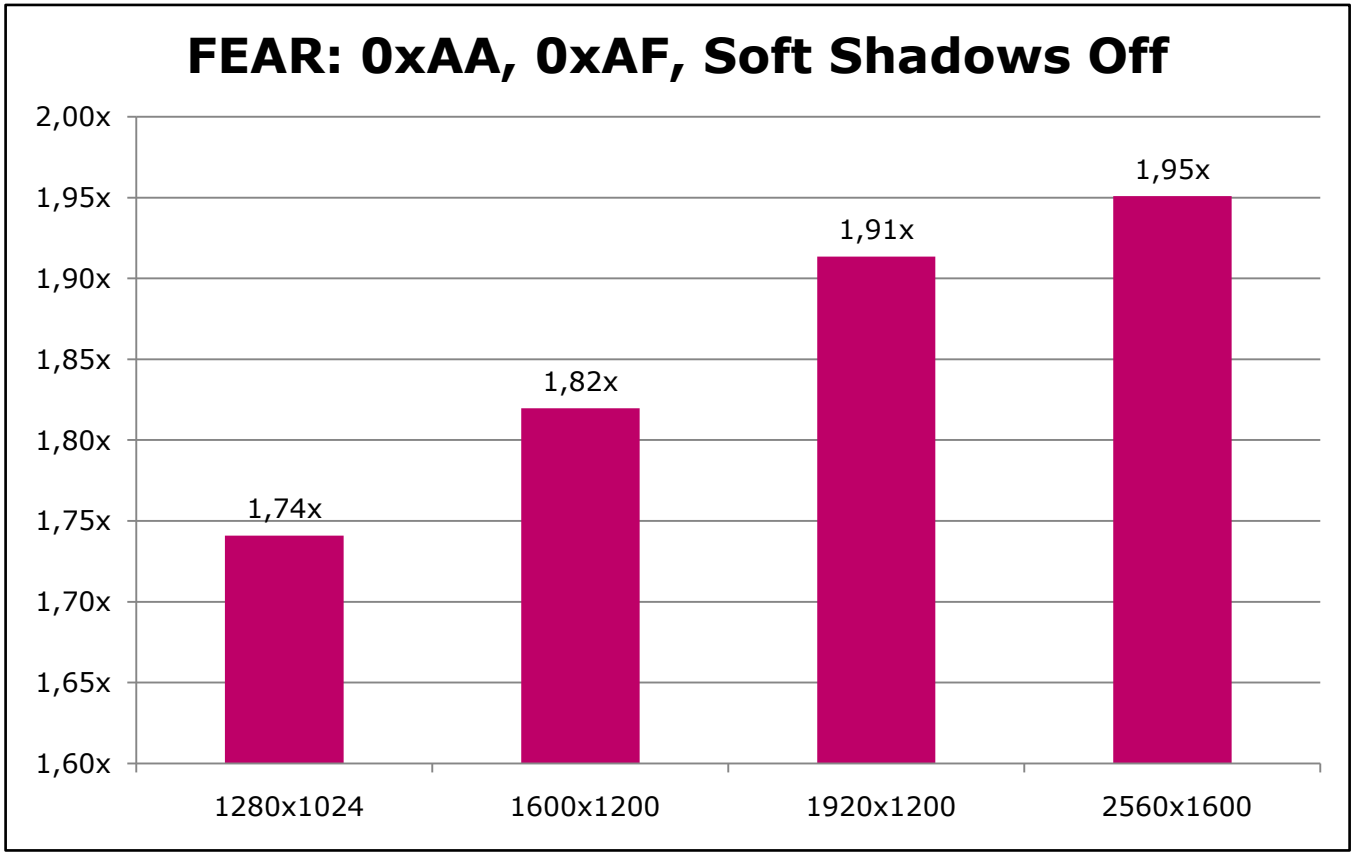
* HD3870 vs HD3870 X2 (QX9650 (3.0/1333), ASUS X38, 2GB DDR2-800, Windows VISTA64)

Half-Life 2



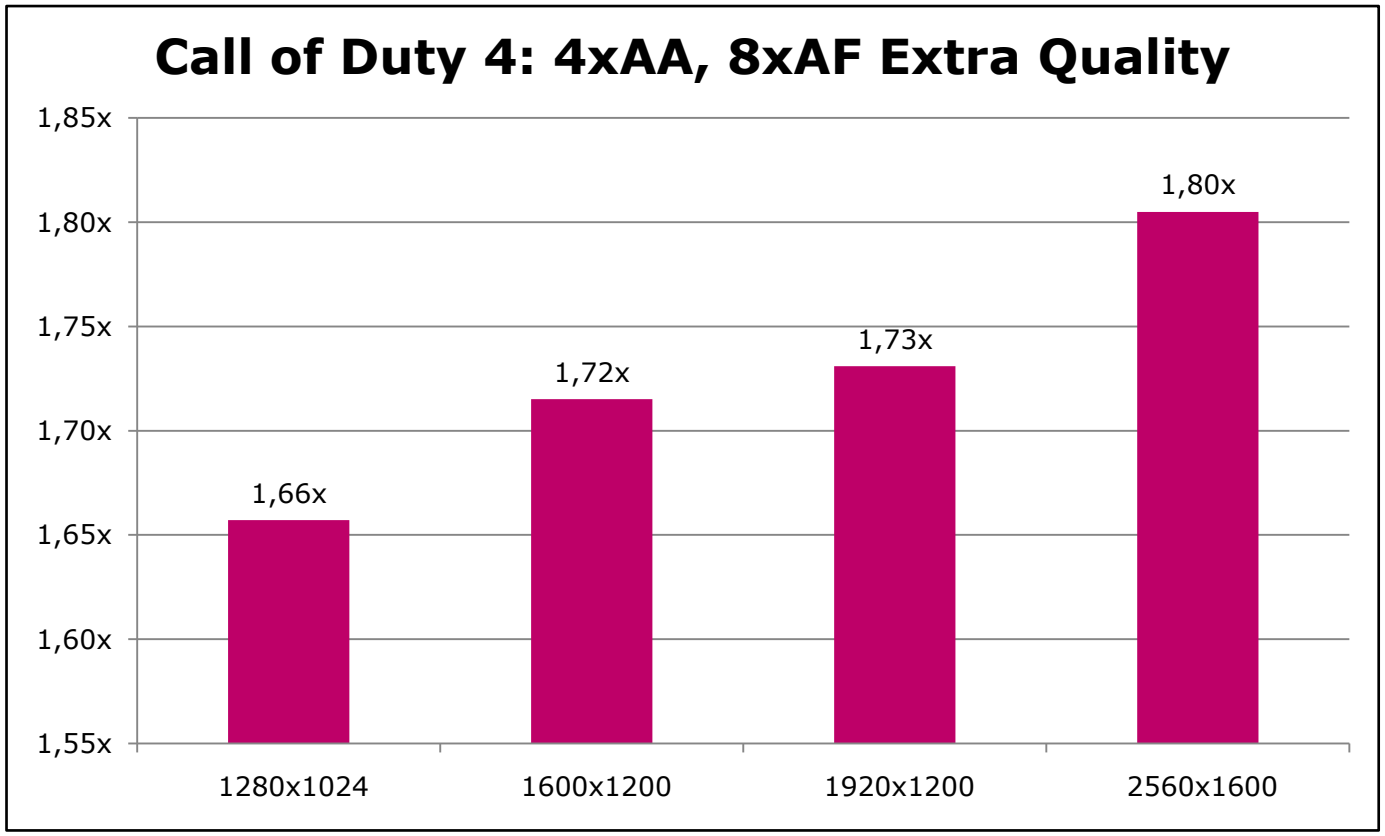
* HD3870 vs HD3870 X2 (QX9650 (3.0/1333), ASUS X38, 2GB DDR2-800, Windows VISTA64)

FEAR



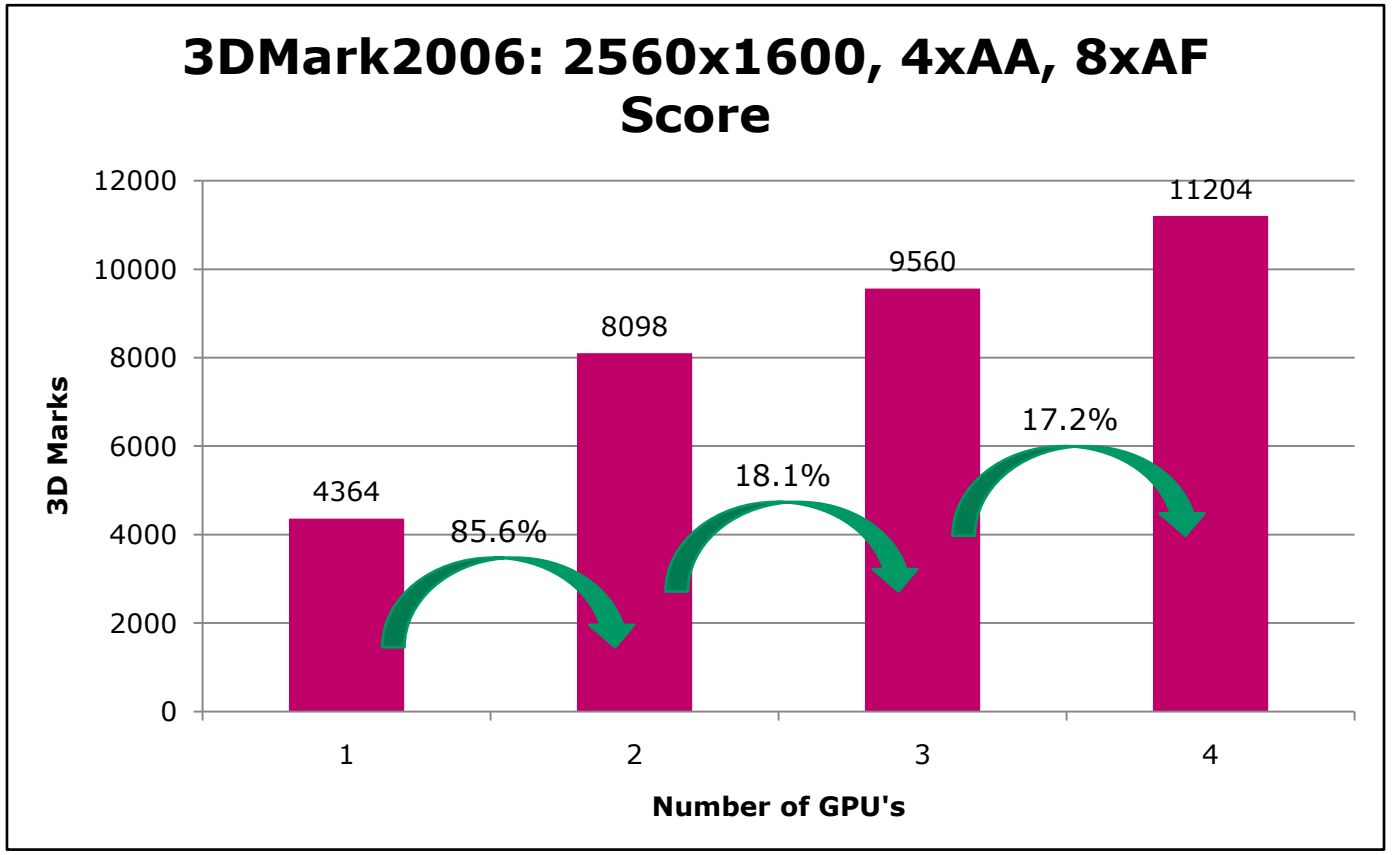
* HD3870 vs HD3870 X2 (QX9650 (3.0/1333), ASUS X38, 2GB DDR2-800, Windows VISTA64)

Call of Duty 4



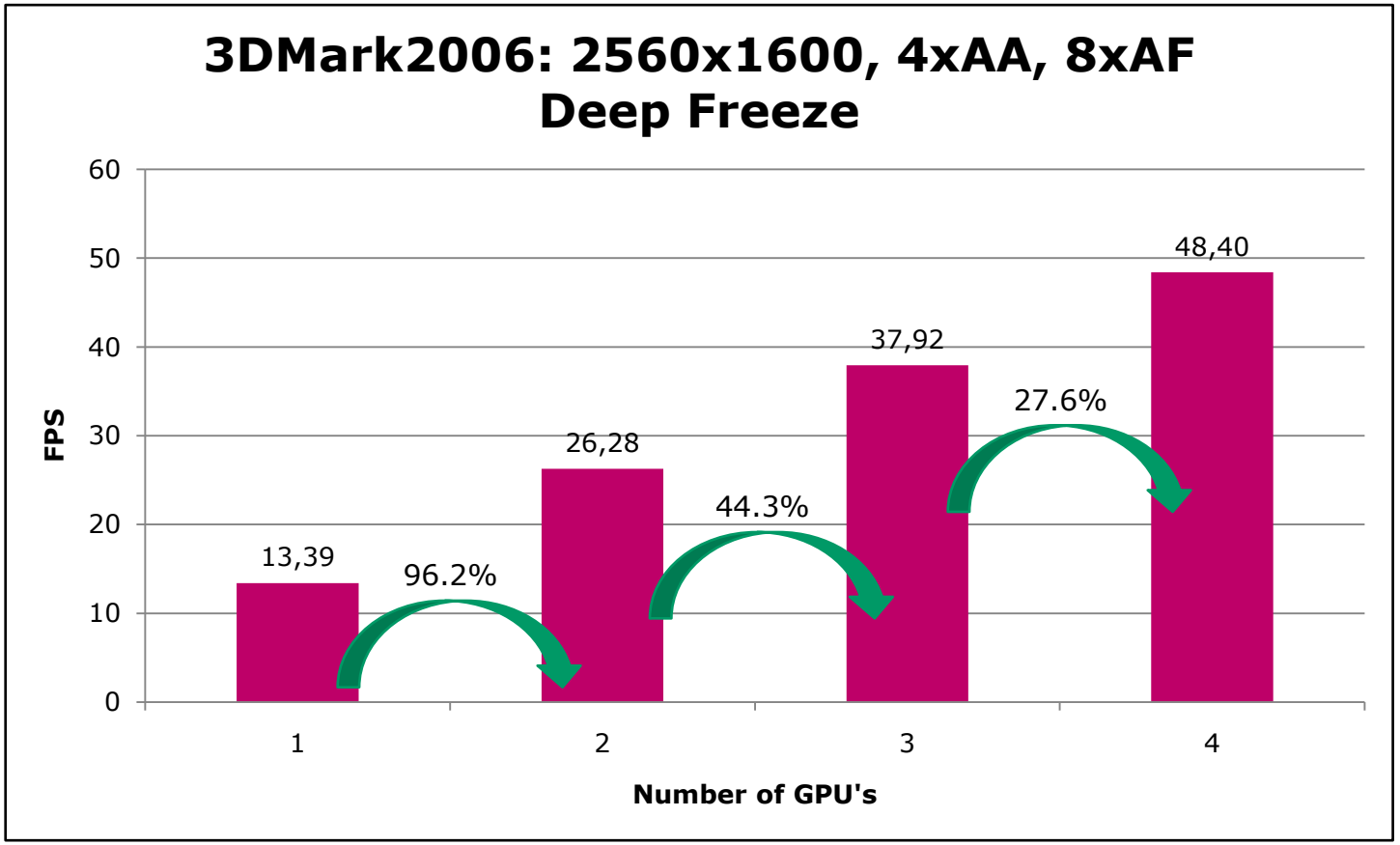
* HD3870 vs HD3870 X2 (QX9650 (3.0/1333), ASUS X38, 2GB DDR2-800, Windows VISTA64)

3DMark2006: 1 - 4 GPUs



* 1-4 HD3870 (Phenom 2.6GHz, MSI K9A2, 2GB DDR2-800, Windows VISTA64)

3DMark2006: 1 – 4 GPUs

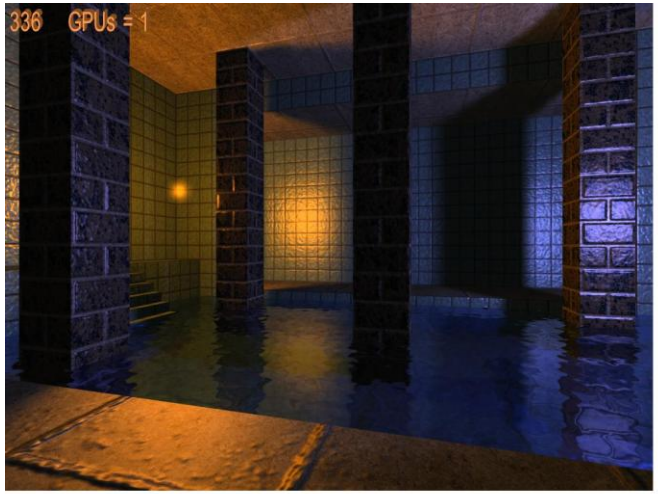


* 1-4 HD3870 (Phenom 2.6GHz, MSI K9A2, 2GB DDR2-800, Windows VISTA64)

Call to Action

AFR-Friendly SDK Sample

- Part of the ATI developer SDK
  - http://ati.amd.com/developer
- Detects the number of GPUs
- Correctly deals with textures used as render targets
- Provides a solution for dealing with mouse cursor lag
- Go and take a look!!



Call to Action

AMD
Smarter Choice

- MGPUs provide demonstrable performance gains
- MGPUs boost visual quality
- Plan from day one to make your rendering scale
  - Detect the number of GPUs
- Regularly check for AFR unfriendly behavior
- Visit http://ati.amd.com/developer
- Talk to us...
  - jon.story@amd.com
  - holger.gruen@amd.com

AMD
Smarter Choice

Please Fill in the Feedback Forms

AMD
Smarter Choice

Questions?